Red Hat Reference Architecture Series

Deploying Highly Available SAP® Servers using Red Hat® Cluster Suite

SAP® NetWeaver based Applications
SAP® NetWeaver 2004 / SAP® NetWeaver 7.0
Oracle 10g / MaxDB
Red Hat® Cluster Suite
Red Hat® Enterprise Linux 5.3

Version 1.0

March 2009

redhat

ATIX

ADVANCED TECHNOLOGY FOR INDIVIDUAL SUCCESS

REALTECH

TECHNOLOGY SAP GLOBAL PARTNER

Deploying Highly Available SAP® Servers using Red Hat® Cluster Suite

1801 Varsity Drive™
Raleigh NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park NC 27709 USA

The following terms used in this publication are trademarks of other companies as follows:

- Linux is a registered trademark of Linus Torvalds
- Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries
- SAP® software, SAP NetWeaver™ platform, SAP® R/3® Enterprise software, mySAP® and all other SAP products and services mentioned herein are registered trademarks of SAP AG in Germany and other countries

All other trademarks referenced herein are the property of their respective owners.

© 2009 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at http://www.opencontent.org/openpub/).

The information contained herein is subject to change without notice. Red Hat, Inc., SAP AG, ATIX AG, and REALTECH AG shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of Red Hat Inc., SAP AG, ATIX AG, and REALTECH AG.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from Red Hat Inc., SAP AG, ATIX AG, and REALTECH AG.

The GPG fingerprint of the security@redhat.com key is:
CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Table of Contents

1 Executive Summary....7
1.1 Introduction....7
1.2 Audience....8
1.3 Acronyms....8
1.4 Reference Documentation....9
1.5 SAP Overview....10
1.6 Cluster Technology Overview....11
2 Testbed Environment....11
3 Hardware Requirements....12
3.1 Shared Storage Requirements....12
3.2 Server Hardware Requirements....12
3.3 Network Requirements....13
4 Red Hat Cluster Basics....13
4.1 OpenAIS....13
4.2 CMAN....13
4.3 Quorum....13
4.4 Qdisk....14
Additional heuristics....15
4.5 GFS....15
4.6 DLM....16
4.7 Fencing....16
4.7.1 Power Fencing Systems....16
4.7.2 SAN Based Fencing....16
4.7.3 SCSI Fencing 16
4.8 CLVM....17
4.9 Storage Mirroring....17
4.10 Cluster Resource Manager....18
4.11 Shared Root Cluster....18
4.12 Scalability....18
4.13 Availability....18
4.14 Management....18
5 Operating System Installation....19
5.1 OS Customizations....19
5.1.1 NTP....19

5.1.2 ACPI....19
5.1.3 Firewall....20
5.2 Network Configuration....20
5.2.1 Public/Private Networks....20
5.2.2 Bonding....20
5.2.3 Hosts file....21
5.3 Storage Configuration....21
5.3.1 Multipathing....21
5.3.2 Device Mapper Multipath....21
5.3.3 CLVM....21
5.3.3.1 LVM Configuration....22
5.3.3.2 Volume Configuration....22
5.3.4 GFS....22
5.3.4.1 Formatting....22
5.3.4.2 fstab....23
5.4 Cluster Software Installation....23
5.4.1 Core Cluster Software....23
5.4.2 Open-Sharedroot Software....23
5.5 Cluster Core Configuration....24
5.5.1 CMAN / OpenAIS....25
5.5.2 Qdisk....25
5.5.3 Fencing....27
5.5.4 Cluster Nodes....27
5.6 Local root Cluster Installation....28
5.7 Shared Root Cluster Installation....28
6 SAP Installation....29
6.1 Local Root and Shared Storage with Local File System Types....29
6.1.1 SAP Architecture....29
6.1.2 SAP Virtual IP Addresses....30
6.1.3 SAP File Systems....30
6.1.3.1 Local File Systems....30
6.1.3.2 Shared Storage File Systems....30
6.1.3.3 NFS Mounted File Systems....31
6.1.3.4 Before Starting the SAP Installation....31
6.1.4 Installation with sapinst....31
6.1.5 Installation Post-Processing....31
6.1.5.1 Users, Groups and Home Directories....31
6.1.5.2 Synchronizing Files and Directories....32
6.1.5.3 SAP Profiles....32

6.1.5.4 SAP Release-specific Post-processing....33
6.1.5.5 Before Starting the Cluster....33
6.1.6 Enqueue Replication Server....33
6.2 Local Root and Shared Storage with GFS....33
6.2.1 SAP Architecture....33
6.2.2 SAP Virtual IP Addresses....34
6.2.3 SAP File Systems....34
6.2.3.1 Local File Systems....34
6.2.3.2 Shared Storage File Systems....34
6.2.3.3 NFS Mounted File Systems....34
6.2.3.4 Before Starting the SAP Installation....35
6.2.4 Installation with sapinst....35
6.2.5 Installation Post-Processing....35
6.2.5.1 Users, Groups and Home Directories....35
6.2.5.2 Synchronizing Files and Directories....35
6.2.5.3 SAP Profiles....36
6.2.5.4 SAP Release-specific Post-processing....36
6.2.5.5 Before Starting the Cluster....36
6.2.6 Enqueue Replication Server....36
6.3 Shared Root and Shared Storage with GFS....37
6.3.1 SAP Architecture....37
6.3.2 SAP Virtual IP Addresses....37
6.3.3 SAP File Systems....37
6.3.3.1 File Systems on Shared Root....37
6.3.3.2 Shared Storage File Systems....37
6.3.3.3 NFS Mounted File Systems....38
6.3.3.4 Before Starting the SAP Installation....38
6.3.4 Installation with sapinst....38
6.3.5 Installation Post-Processing....38
6.3.5.1 Users, Groups and Home Directories....38
6.3.5.2 SAP Profiles....39
6.3.5.3 SAP Release-specific Post-processing....39
6.3.5.4 Before Starting the Cluster....39
6.3.6 Enqueue Replication Server....39
7 Resource Manager....39
7.1 Cluster Resources....39
7.2 Configuration....40
7.3 Failover Domains....40
7.4 Cluster Resources and Services....42
7.4.1 IP....43
7.4.2 Netfs....43

7.4.3 FS....44
7.4.4 SAPInstance....45
7.4.5 SAPDatabase....49
7.5 Dependencies....52
7.5.1 Resource Dependencies....52
7.5.2 Service Dependencies....53
7.5.2.1 Hard and Soft Dependencies....53
7.5.2.2 Follow Service Dependency....53
8 Cluster Management....54
8.1 CMAN....54
8.1.1 cman_tool status....54
8.1.2 cman_tool nodes....55
8.1.3 cman_tool services....55
8.2 rgmanager....56
8.2.1 clustat....56
8.2.2 clusvcadm....56
8.2.3 rg_test....56
8.3 Open-Sharedroot....57
8.3.1 Host dependent files....57
8.3.2 Update initrd....57
8.3.3 Cluster Clone....57
Appendix A: cluster.conf....58
Appendix B: multipath.conf....60
Appendix C: lvm.conf....63

1 Executive Summary

This paper details the deployment of a highly available SAP service on a Red Hat Enterprise Linux 5 cluster. After an introduction to the basic concepts and system requirements, this document will provide detailed information about the Red Hat Cluster Suite (RHCS), SAP NetWeaver, and cluster configuration options.

1.1 Introduction

A cluster is essentially a group of two or more computers working together which, from an end user's perspective, appear as one server. Clustering can be used to enable storage clustering, balance load among cluster members, parallel processing, and high-availability (HA). The "highly available" aspect of any cluster service indicates that it is configured in a manner such that the failure of any one cluster member, or a subsystem failure within a member, will not prevent the continued availability of the service itself.

Ensuring the highest possible availability of your SAP systems is essential for success. The availability of an SAP application typically depends on an SAP application server which in turn relies on optimal availability of an underlying database. This layered software stack sits atop an even more complex hardware infrastructure. In order to increase the availability of SAP software, redundant hardware and additional cluster software is required. The cluster software monitors the status of services provided by running SAP software and initiates failover to redundant server infrastructure if problems are detected. RHCS, bundled with Red Hat Enterprise Linux, provides the features necessary to make critical SAP services highly available. This document will illustrate the recommended highly available RHCS infrastructure for SAP.

When configuring HA for an SAP environment, all the software and hardware layers must be taken into consideration. Red Hat, ATIX and REALTECH , with guidance from SAP, have conducted development work in order to provide a reference architecture of High Availability for SAP NetWeaver using RHCS. This implementation is compliant with SAP recommendations and this document is the result of this partnered development. Together with the resource scripts in Red Hat Enterprise Linux 5.3, this reference architecture can serve as a guide to deploying highly available SAP applications (the majority of the SAP product portfolio including ECC, SCM, SRM, etc.) based upon SAP NetWeaver technology. It is provided as is without support or liability statements. Experts at the required technologies may follow this guide when implementing highly available, Red Hat Enterprise Linux based SAP system(s).

Note that cluster software has full control of all services including the starting and stopping of SAP software. Numerous customer cases have proven how a poorly configured HA environment can inadvertently decrease the availability of critical SAP systems. As such, consulting services familiar with both SAP and RHCS would be a cost effective investment.

1.2 Audience

This document addresses SAP certified technical consultants for SAP NetWeaver with experience in HA systems. Access to SAP information resources such as SAP Marketplace is mandatory.

1.3 Acronyms

Common acronyms referenced within this document are listed below.

AAS	SAP Additional Application Server
ADA	SAP Database Type MaxDB
API	Application Programming Interface
ASCS	SAP ABAP Central Services Instance
CLVM	Cluster Logical Volume Manager
CMAN	Cluster Manager
DB6	SAP Database Type DB2 on Linux
DLM	Distributed Lock Manager
ERS	SAP Enqueue Replication Server
ERS	Enqueue Replication Server
GFS	Global File System
HA	High-Availability
IP	Internet Protocol
NAS	Network Attached Storage
NFS	Network File Server
NIC	Network Interface Card
NTP	Network Time Protocol
NW640	SAP NetWeaver 2004 (kernel 6.40)
NW70	SAP NetWeaver 7.0
OCF	Open Cluster Framework
ORA	SAP Database Type Oracle
OS	Operating System
PAS	SAP Primary Application Server
POSIX	Portable Operating System Interface
QDISK	Quorum Disk
QDISKD	Quorum Disk Daemon

RHCS	Red Hat Cluster Suite
RHEL	Red Hat Enterprise Linux
RIND	Rind Is Not Dependencies
SAN	Storage Area Network
SCS	SAP Central Services Instance (for Java)
SPOF	Single Point Of Failure
SSI	Single System Image
VFS	Virtual File System

1.4 Reference Documentation

The following list includes the existing documentation and articles referenced by this document.

- **Red Hat Enterprise Linux Installation Guide**

  http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.2/html/Installation_Guide/index.html

- **Configuring and Managing a Red Hat Cluster**

  http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5.2/html/Cluster_Administration/index.html

- **SAP Installation Guide**

  http://service.sap.com/instguides

- **SAP Technical Infrastructure Guide**

  https://www.sdn.sap.com/irj/sdn/ha

- **RHEL5 GFS Shared Root Mini Howto**

  http://www.open-sharedroot.org/documentation/rhel5-gfs-shared-root-mini-howto

- **RHEL5 NFS Shared Root Mini Howto**

  http://www.open-sharedroot.org/documentation/rhel5-nfs-shared-root-mini-howto

- **Open-Sharedroot Administrators Handbook**

  http://www.open-sharedroot.org/documentation/administrators-handbook

- **“Data sharing with a Red Hat GFS storage cluster”**

  http://www.redhat.com/magazine/021jul06/features/gfs_update/

- **“Enhancing cluster quorum with Qdisk”**

  http://magazine.redhat.com/2007/12/19/enhancing-cluster-quorum-with-qdisk/

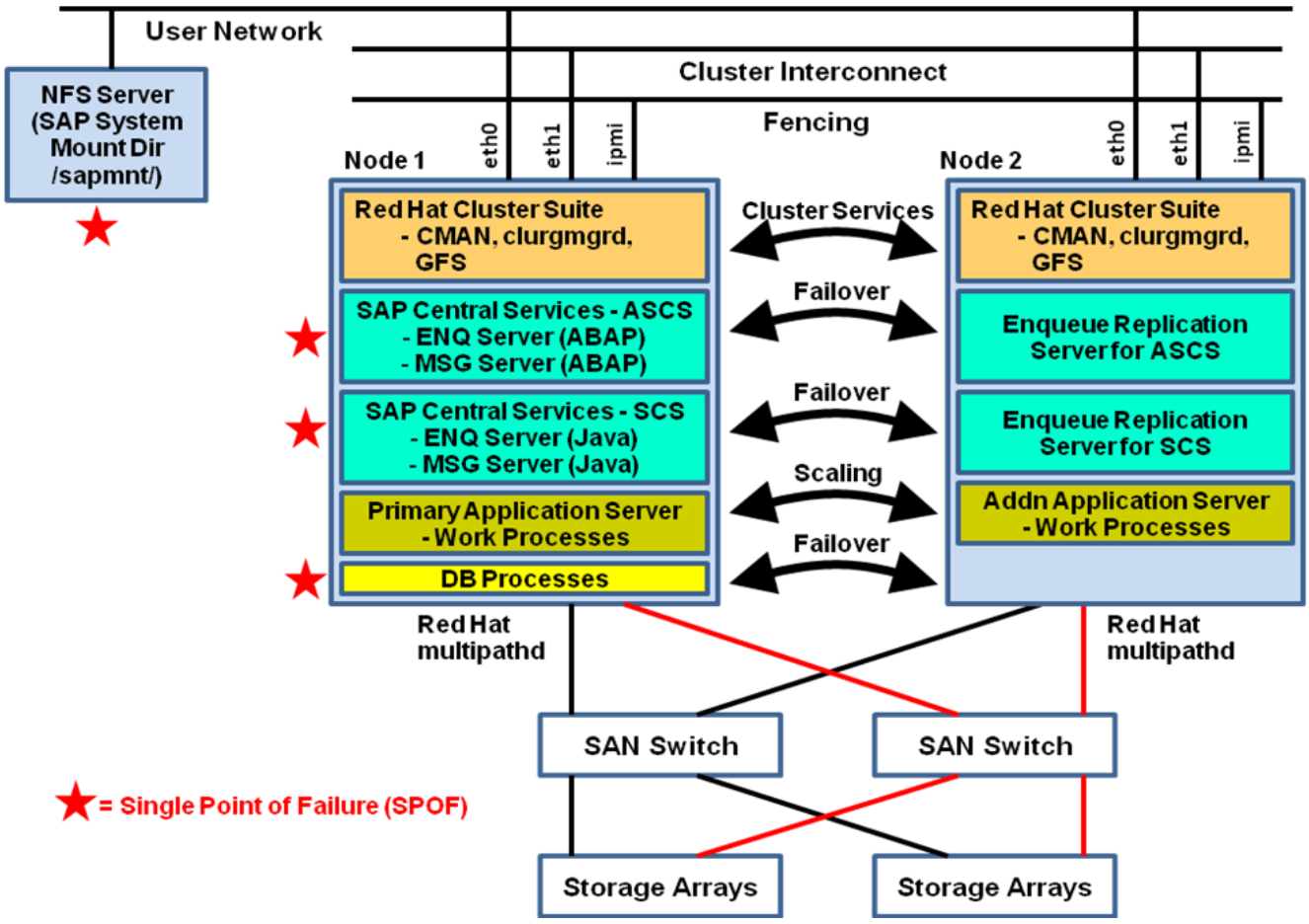
- **“Failover Domains“**

  http://sources.redhat.com/cluster/wiki/FailoverDomains

1.5 SAP Overview

In an SAP NetWeaver environment, these single points of failure (SPOF) must be considered:

- Database
- SAP Central Services Instance (SCS/ASCS)
- SAP System Mount Directory (*/sapmnt/<SID>*)



The SAP example system in the above illustration is a double stack with Enqueue Replication, both for ASCS and SCS. Although they are not SPOF, the Enqueue Replication Servers (ERS) are controlled by the cluster software. To create a working Enqueue Replication, it is important that the ERS does not run on the same cluster node as the (A)SCS. This is because the original enqueue table lies within the same shared memory segment as the replicated table.

When the (A)SCS fails and a failover is triggered by the cluster software, (A)SCS must start on the node where the ERS is running. When (A)SCS comes up, it shuts down the ERS instance without cleaning its shared memory and attaches itself to the shared memory segment where the ERS had stored the replicated enqueue table. Now the replicated enqueue table has become the new original enqueue table.

To ensure that the (A)SCS "follows" the ERS instance, the follow-service dependency was implemented in RHCS.

The SAP System Mount Directory should be exported by a highly available NFS server and mounted by the cluster software.

1.6 Cluster Technology Overview

For applications that require maximum system uptime, a Red Hat Enterprise Linux cluster with RHCS is the solution. Specifically designed for Red Hat Enterprise Linux, RHCS provides two distinct types of clustering:

- Application/Service Failover - Create n-node server clusters for failover of key applications and services
- IP Load Balancing - Load balance incoming IP network requests across a farm of servers

With RHCS, applications can be deployed in HA configurations so that they are always operational, bringing "scale-out" capabilities to Enterprise Linux deployments.

RHCS provides a complete, ready-to-use failover solution for SAP NetWeaver.

2 Testbed Environment

This section provides information about the hardware and software used to build the highly available SAP system.

Operating System:

- Red Hat Enterprise Linux 5.3 including:
  - Latest updates via Red Hat Network (RHN) subscription to channel(s):
    - Red Hat Enterprise Linux
  - Resolutions to the following Bugzilla issues:
    - 475828
    - 481762
    - 485026

Cluster Software:

- RHCS 5.3 including:
  - Latest updates via RHN subscriptions to channel(s):
    - RHEL Clustering
    - RHEL Cluster-Storage
- Open-Sharedroot 4.4 with latest available patches

Hardware:

- Cluster Servers:
  - (2) Fujitsu Siemens Rx220
- Storage:
  - EMC CLARiiON
- SAN Infrastructure:
  - QLogic

SAP Installation:

- SAP NetWeaver 2004, WebAS ABAP on MaxDB
- SAP NetWeaver 7.0, WebAS ABAP+JAVA on Oracle

3 Hardware Requirements

3.1 Shared Storage Requirements

Shared storage indicates external storage accessible by every cluster member. For increased availability, the Storage Area Network (SAN) is configured with multiple physical paths between servers and storage in an attempt to reduce any single point of cluster failure. Whether using iSCSI, fibre channel or another means of connecting to shared storage, higher availability is typically accomplished by using:

- multiple HBAs on the hosts
- multiple switches (if applicable) between host and storage
- multiple controllers on the storage arrays
- RAID sets when creating LUNs on the storage arrays
- LVM mirroring of volumes across storage arrays

In this manner, the chances of any one hardware failure resulting in a loss of cluster functionality is greatly reduced.

Because there are many varying storage vendors available, this document assumes that LUNs have been created on the storage and presented to all HBAs on each host for subsequent device discovery.

3.2 Server Hardware Requirements

The server hardware requirements can be fulfilled by almost any enterprise grade server. The supported architectures are single or multicore x86_64 processors. Typically, SAP servers are equipped with a fair amount of memory starting at 2 gigabytes and are usually limited only by the hardware specification. The cluster nodes need to be attached to a fencing mechanism. Please reference the "Cluster Fencing“ section for further information.

3.3 Network Requirements

There should be at least two Network Interface Cards (NIC), whether embedded or added to each server. Where multiple network interfaces are available, NIC bonding can be implemented for additional availability and is the only current method providing a NIC failover ability. One bonded interface will be configured with an external IP address while the other will be configured as an interconnect between cluster members using local network connectivity. Clusters are very dependent on constant communication between nodes which are maintained across the local interconnect. It is highly recommended that a private network be used for all intra-cluster communication.

The Red Hat cluster infrastructure in Red Hat Enterprise Linux 5 uses multicast. Some network switches require special configuration settings for a well functioning multicast operation. Please refer to the hardware vendor's configuration guide for correct multicast configurations.

4 Red Hat Cluster Basics

4.1 OpenAIS

In Red Hat Enterprise Linux 5, the core cluster infrastructure is based on the OpenAIS framework. OpenAIS is an open source implementation of the Application Interface Specification defined by the Service Availability Forum, based upon extended virtual synchrony. The project currently implements Application Programming Interfaces (API) for application failover, application defined checkpointing, application eventing, extended virtual synchrony, and cluster membership.

Today, OpenAIS is the state of the art, core cluster framework included in most Enterprise Linux distributions.

The heart of OpenAIS is the *aisexec* daemon, into which various services are loaded.

OpenAIS uses multicast for the internal cluster communication.

4.2 CMAN

Cluster Manager (CMAN) is a Red Hat specific service module that loads in the OpenAIS daemon. It provides a user API that is used by Red Hat layered cluster components.

CMAN also provides additional functionality such as APIs for a quorum disk, the quorum itself, conditional shutdown, and barriers.

4.3 Quorum

A cluster typically uses shared data resources such as a cluster file system (e.g., GFS) or local file systems controlled by the cluster resource management system (e.g., rgmanager). The cluster must be aware of the current state of the shared resources at all times. Therefore, it must be guaranteed that every critical transition within the cluster cannot compromise data integrity.

In the event of a major network problem, cluster partitioning (aka: split-brain situation) can occur. Each partition can no longer communicate with nodes outside its own partition. A Red Hat cluster requires the *quorum requirement* be fulfilled before a status change in the cluster is allowed. For example, quorum is required by the resource management system to relocate cluster resources or for the CMAN module to remove nodes from the cluster.

The cluster partition is considered quorate if more than half of all votes within the entire cluster belong the the cluster partition.

$$Q = V/2 + 1$$

where Q is the required number of votes for quorum and V is the total number of votes within the cluster.

Although the quorum requirements calculation based on the active nodes in a cluster work well for various cluster configurations and network issues, specific cases exist where the cluster cannot decide or incorrect decisions have been made.

As such, the use of a quorum disk (qdisk) has been reintroduced.

4.4 Qdisk

In specific cases, the quorum requirement based on the nodes belonging to a cluster is insufficient. In a two node cluster, the standard quorum calculation (Q = V/2 +1) would result in two, considering one vote per cluster node. In the case of a highly available cluster, this would make no sense. Therefore, the two node cluster is considered a special case and by using the *<two_node>* configuration option, quorum is reduced to one. In this manner, quorum is maintained and if one node should fail, the other will take over on its own.

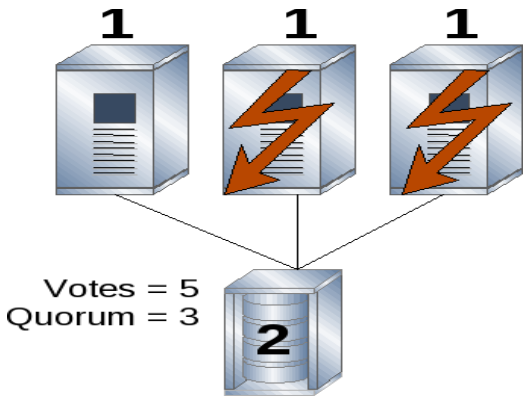
One concern with this solution is in the case of a network loss between nodes, each node will interpret the lack of connectivity as a failure of the other node. This problem is most commonly referred as a “split brain“ situation. As each node assumes it is the survivor, it will attempt to fence the other node to prevent uncontrolled access to the shared storage resources. In this instance, which ever node successfully fences the other first will become the surviving member.

A quorum disk (qdisk) can be used to prevent this situation, bolstering the quorum by adding an additional vote or votes to the cluster.

In a two node cluster configuration with a qdisk, the total expected votes would be three with a quorum of two.

In small multi-node cluster configurations, other types of problems can occur. In a three or four node cluster, quorum is two or three respectively, and losing two nodes can cause a problem.

To resolve the small cluster quorum problem, a quorum disk with a vote count equaling the number of cluster nodes minus one bolsters the quorum enough to enable the cluster to survive with only one node remaining.



The quorum disk daemon (qdiskd) runs on each node in the cluster, periodically evaluating its own health and then placing its state information into an assigned portion of the shared disk area. Each qdiskd then looks at the state of the other nodes in the cluster as posted in their area of the QDisk partition. When in a healthy state, the quorum of the cluster adds the vote count for each node plus the vote count of the qdisk partition. In the above example, the total vote count is five; one for each node and two for the qdisk partition.

If, on any node, qdisk is unable to access its shared disk area after several attempts, then the qdiskd on another node in the cluster will attempt to fence the troubled node to return it to an operational state.

Additional heuristics

Red Hat adds an additional feature to the quorum disk mechanism. Optionally, one or more heuristics can be added to the qdisk configuration. Heuristics are tests performed by the qdisk daemon to verify the health of the node on which it runs. Typical examples are verifications of network connectivity such as the server's ability to ping network routers. Heuristics can also be used to implement network tiebreaker functionality.

4.5 GFS

Red Hat's Global File System (GFS) is a POSIX compliant, symmetric shared cluster file system. GFS lets servers share files with a common file system on a SAN.

With local file system configurations such as ext3, only one server can have access to a disk or logical volume at any given time. In a cluster configuration, this approach has two major drawbacks. First, active/active file system configurations cannot be realized, limiting scale out ability. Second, during a failover operation, a local file system must be be unmounted from the server that originally owned the service and must be remounted on the new server.

GFS creates a common file system across multiple SAN disks or volumes and makes this file system available to multiple servers in a cluster. Scale out file system configurations can be easily achieved. During a failover operation, it is not necessary to unmount the GFS file system because data integrity is protected by coordinating access to files so that reads and writes are consistent across servers. Therefore, availability is improved by making the file system accessible to all servers in the cluster. GFS can be used to increase performance, reduce management complexity, and reduce costs with consolidated storage resources.

GFS runs on each node in a cluster. As with all file systems, it is basically a kernel module that runs on top of the Virtual File System (VFS) layer of the kernel. It controls how and where the data is stored on a block device or logical volume. In order for cluster members to cooperatively share the data on a SAN, GFS is required to coordinate a distributed locking protocol.

4.6 DLM

The Distributed Lock Manager (DLM) is a cluster locking protocol in the form of a kernel module. Its ensure that clustered nodes that share storage do not corrupt shared data. DLM provides a high performance locking mechanism required by the Cluster Logical Volume Manager (CLVM).

4.7 Fencing

Fencing is essentially the act of isolating a node in a cluster when it is deemed malfunctioning or otherwise unresponsive, effectively ensuring that it can no longer perform I/O to shared storage. Fencing typically occurs automatically in order to protect processes on another active node from modifying the resources during node failures.

Fencing is required because it is impossible to distinguish a true failure from a temporary hang. If the malfunctioning node is truly down, then it cannot do any damage and so theoretically no action would be required (it could simply be brought back into the cluster with the usual join process). However, because there exists the possibility that a malfunctioning node could believe that the remaining cluster members are themselves malfunctioning, a race condition could ensue with the possibility of data corruption. Instead, the system must assume the worst and fence in case of problems.

4.7.1 Power Fencing Systems

The power fencing subsystem allows operational cluster nodes to control the power of failed nodes to ensure that they do not access storage in an uncoordinated manner. Most power control systems are network based. They are available from system vendors as add-in cards or integrated into the motherboard. External power fencing devices are also available. These are typically rack or cabinet mounted units into which servers are plugged.

4.7.2 SAN Based Fencing

While it is preferable to employ a power fencing solution for the robustness a system reboot provides, SAN switch fencing is also possible. As with Power Fencing, the need is to protect shared data. SAN switch fencing works by preventing access to storage at the SAN switch.

4.7.3 SCSI Fencing

SCSI persistent reservations can be used for I/O fencing. All nodes in the cluster must register with the SCSI device to be able to access the storage. If a node has to be fenced, the registration is revoked by the other cluster members.

Reference the *fence_scsi(8)* manpage for further details

Please note, that the SCSI fencing mechanism requires persistent SCSI reservations. Please contact Red Hat technical support and your storage hardware vendor if your software and hardware configuration supports persistent SCSI reservations.

4.8 CLVM

Consistency must be ensured in all cluster configurations. Logical volume configurations are protected by the use of CLVM. CLVM is an extension to standard Logical Volume Management (LVM) that distributes LVM metadata updates to the cluster. The CLVM daemon (clvmd) must be running on all nodes in the cluster and will produce an error if any node in the cluster does not have this daemon running.

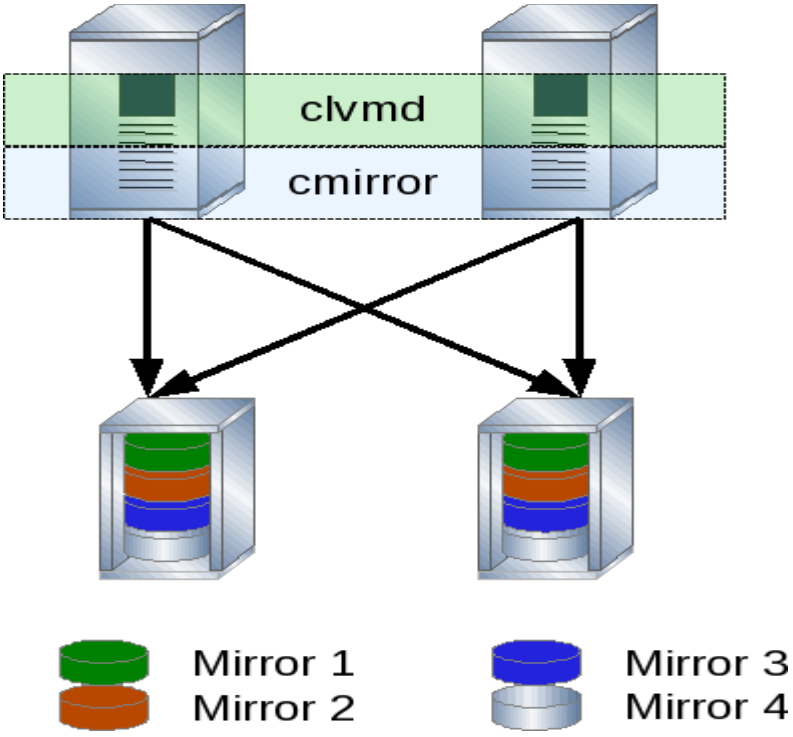
4.9 Storage Mirroring

In disaster tolerant configurations, storage mirroring techniques are used to protect data and ensure availability in the event of a storage array loss. Storage mirroring is normally performed in two different ways.

Enterprise storage arrays typically offer the mechanism to mirror all data from one storage array to one or more other arrays. In the case of a disaster, remote data copies can be used.

In Red Hat Enterprise Linux 5.3, Red Hat offers the possibility to create cluster aware host based mirroring configurations. With the *cmirror* device mapper plugin, all servers in the cluster are able to create mirrors of independent storage arrays in order to prevent data loss and ensure availability in case of disaster.

CLVM is used to manage and monitor the mirrored logical volumes.



On a clustered volume group, the following command can be used to create a cluster aware mirror:

```
# lvcreate -m1 -L 1G -n my_new_lv my_vg
```

4.10 Cluster Resource Manager

The Cluster Resource Manager (rgmanager) manages and provides failover capabilities for cluster resource groups. It controls the handling of user requests including service start, restart, disable, and relocate.

The service manager daemon also handles restarting and relocating services in the event of failures. Rgmanager uses Open Cluster Framework (OCF) compliant resource agents to control and monitor required resources. *SAPInstance* and *SAPDatabase* are OCF compliant resource agents provided by Red Hat.

In Red Hat Enterprise Linux 5.3, rgmanager includes an event driven scripting mechanism called RIND (Rind Is Not Dependencies). RIND can be used to create complex event driven service dependencies. For automatic enqueue replication failover scenarios, the RIND based *follow_service* dependency is required.

4.11 Shared Root Cluster

Modern data center concepts are treating server and storage hardware independently. SAN or Network Attached Storage (NAS) devices are used to store and protect all critical data. Hence, servers are no longer tied to a single operating system installation and can be used in more flexible manners. All servers are booted directly from the shared storage arrays.

A single cluster wide OS installation can be placed upon a shared file system such as NFS or GFS. The installation can easily be shared by all servers in a cluster. In this manner, a diskless, shared root cluster with a file system based Single System Image (SSI) is achieved.

The open-sharedroot cluster enhancement must be installed to create a Red Hat Enterprise Linux based diskless shared root cluster.

4.12 Scalability

Within a diskless shared root cluster, the server scaling is independent from storage scaling. If a diskless shared root cluster needs more CPU resources one simply needs to add new servers on the fly.

4.13 Availability

In a shared root cluster, cluster wide operating system configuration consistency is assured and required to ensure reliable failover mechanisms.

4.14 Management

One of the key advantages of a diskless shared root cluster is the ease of management.

Compared to other cluster concepts, the management and operation is straightforward. With classic application clusters, the complexity of management is proportional to the number of nodes in the cluster. Any change must be rolled out on every node.

With a diskless shared root cluster, one can change information on any node and the change will be observed by all nodes automatically. No error-prone replication processes are required to submit changes on any node. Software updates can be made on one node and all other nodes will immediately benefit from the changes. This ensures that all cluster nodes are equal and any node can become a single point of control.

Diskless shared root cluster configurations reduce the complexity of cluster configurations to that of local servers.

5 Operating System Installation

Reference the Red Hat Enterprise Linux Installation Guide for the specific details regarding the acquisition and installation of Red Hat Enterprise Linux. The guide includes information specific to the platform on which the installation will take place (x86, AMD64, Intel® 64 and Itanium) so be sure to read the appropriate section for your platform.

Once the platform specific information has been understood and the hardware configuration has been performed to accommodate a cluster, install Red Hat Enterprise Linux 5.3 on the server(s) using the preferred method.

The install process will guide the user through the OS installation. The Red Hat Enterprise Linux Installation Guide will provide details regarding each of the screens that will be presented during the installation process.

Please refer to the Installation section of Configuring and Managing a Red Hat Cluster to make sure the required cluster software packages will be installed.

If a local root installation is preferred, the installation process must be performed on each cluster member whereas for a shared root configuration, only one server need be installed.

5.1 OS Customizations

5.1.1 NTP

The synchronization of system clocks in a cluster becomes infinitely more important when storage is shared among members. System times can by synchronized to a network time server via the Network Time Protocol (NTP) by using the ntpd service.

5.1.2 ACPI

Please reference the *Configuring ACPI For Use with Integrated Fence Devices* section in Configuring and Managing a Red Hat Cluster. As described there, disabling ACPI Soft-Off allows an integrated fence device to shut down a server immediately rather than attempting a clean shutdown. Soft-Off allows some components to remain powered so the system can be roused from input from the keyboard, clock, modem, LAN, or USB device and subsequently takes longer to fully shutdown. If a cluster member is configured to be fenced by an integrated

fence device, disable ACPI Soft-Off for that node. Otherwise, if ACPI Soft-Off is enabled, an integrated fence device can take four or more seconds to turn off a node (refer to note that follows). In addition, if ACPI Soft-Off is enabled and a node panics or freezes during shutdown, an integrated fence device may not be able to power off the node. Under those circumstances, fencing is delayed or unsuccessful. Consequently, when a node is fenced with an integrated fence device and ACPI Soft-Off is enabled, a cluster recovers slowly or requires administrative intervention to recover.

```
# chkconfig acpid off
```

5.1.3 Firewall

If use of a local firewall on the cluster nodes is intended, the specific IP ports for the following services must be enabled in order to accommodate RHCS communication requirements:

Service	IP Ports
openais	5404, 5405
rgmanager	41966, 41967, 41968, 41969
ricci	11111
dlm	21064
ccsd	50006, 50007, 50008, 50009

5.2 Network Configuration

In a cluster configuration, the configuration of the cluster interconnect is extremely important. The interconnect is responsible for all internal cluster communication. With a clustered file system, all distributed locking messages are routed through the cluster interconnect. As such, it is highly recommended that the network be reliable and high speed.

5.2.1 Public/Private Networks

At least two network interfaces are recommended for clustering. The reason for this is to separate cluster traffic from all other network traffic. Availability and cluster file system performance is dependent on the reliability and performance of the cluster communication network (private network).

Therefore, all public network load must be routed through a different network (public network).

5.2.2 Bonding

In high availability configurations, at least the private or cluster interconnect network setup, preferably both, must be fully redundant. Network Interface Card (NIC) bonding is the only method to provide NIC failover for the cluster communication network.

5.2.3 Hosts file

The */etc/hosts* file for each cluster member should contain an entry defining localhost. If the external host name of the system is defined on the same line, the host name reference should be removed.

Additionally, each */etc/hosts* file should define the local interconnect of each cluster member.

5.3 Storage Configuration

5.3.1 Multipathing

Storage hardware vendors offer different solutions for implementing a multipath failover capability. This document focuses on the generic multipath device mapper approach. Please consult your storage hardware vendor for the correct and supported multipath configuration.

5.3.2 Device Mapper Multipath

The device mapper multipath plugin (DM multipath) provides greater reliability and performance by using path failover and load balancing. In HA scenarios, cluster servers can use multiple paths to the shared storage devices. Normally these devices are presented as multiple device files (/dev/sdXX)

DM-Multipath creates a single device that routes I/O to the underlying devices according to the multipath configuration. It creates kernel block devices (/dev/dm-*) and corresponding block devices (with persistent names) in the /dev/mapper directory.

The multipath configuration file can also be used to set storage specific attributes. These multipath specific settings are usually obtained from the storage vendor and typically supersede the default settings.

5.3.3 CLVM

In RHCS, LVM managed shared storage must be controlled by High Availability resource manager agents for LVM (HA-LVM) or the clustered logical volume manager daemon (clvmd/ CLVM). Single instance LVM must not be used for shared storage, as it is not cluster aware and can result in data corruption. In this document, CLVM will be used as it allows active/active storage configurations. This allows the use of GFS and secondly failover scenarios are handled more easily.

Note that the qdisk partition cannot be managed by CLVM as this would overwrite the quorum label assigned to the device.

When using CLVM, the clvmd must be running on all nodes. This can be accomplished by enabling the clvmd init script. Please note, that the core cluster must be up and running, before clvmd can be started.

```
# chkconfig clvmd on
# service clvmd start
```

5.3.3.1 LVM Configuration

The LVM configuration file */etc/lvm/lvm.conf* must be modified to enable the use of CLVM.

1. By default, the LVM commands scan all devices found directly in the */dev* path. This is insufficient in dm-multipath configurations.

   There are two ways to enable multipath devices for LVM. The easiest is to modify the *scan* array in the configuration file as follows:

   ```
   scan = [ "/dev/mapper", "/dev/cciss" ]
   ```

2. All changes to logical volumes and their states are communicated using locks. LVM defaults to local file based locking and needs to instead use built-in cluster wide locking. LVM commands are instructed to communicate with the CLVM daemon (clvmd) by the following setting in the global section of the configuration file:

   ```
   locking_type = 3
   ```

   which is normally set accordingly when the cluster is created and the option to use shared storage is selected.

Please reference the LVM Administrators Guide for more detailed information on using LVM in a cluster.

5.3.3.2 Volume Configuration

In this configuration, several storage LUNs were defined to store application data. Each LUN in used for one logical volume group and one logical volume. The following steps were performed to create the clustered logical volume */dev/vg_sap700_gfs/lv_ci*

```
# pvcreate /dev/mapper/DGC_017p1
# vgcreate -cy vg_sap700_gfs /dev/mapper/DGC_017p1
# lvcreate -n lv_ci -L 4G  vg_sap700_gfs
# vgchange -ay g_sap700_gfs
```

5.3.4 GFS

In Red Hat Enterprise Linux 5.3, two versions of the GFS file systems (GFS and GFS2) are available and supported for production use. This document describes the use of GFS.

5.3.4.1 Formatting

To format a storage volume with GFS, the helper tool *gfs_mkfs* must be used. The following options must be defined:

1. **-j number**: number of journals. At least one journal (per machine that will mount the file system) is required.
2. **-t locktable**: The lock table field. Its clustername:fsname. Clustername must match that in cluster.conf; only members of this cluster are permitted to use this file system. Fsname is a unique file system name used to distinguish this GFS file system from others.
3. **-p lockproto**: The locking protocol that should be used. Use the DLM locking

mechanism In a cluster setup, lockproto:=**lock_dlm**

Further information about the `gfs_mkfs` option can be obtained from the `gfs:mkfs(8)` man page.

The following example formats the *lv_ci* logical volume with GFS:

```
# gfs_mkfs -j 3 -p lock_dlm -t lsrhc5:700_ci /dev/vg_sap700_gfs/lv_ci
```

5.3.4.2 fstab

In this GFS based setup, the GFS file systems are an integrated part of the operating system; i.e., the file systems are defined in the */etc/fstab* file and mounted on all cluster nodes during the boot process. It is assumed that the GFS file systems are available when the cluster resource manager service is started.

To integrate a GFS file system into the boot process, add the following line to the */etc/fstab* file:

```
/dev/vg_data/lv_data  /data  gfs  noatime,nodiratime 0 0
```

5.4 Cluster Software Installation

There are multiple ways to install the Red Hat cluster software. In this setup, the following installation procedure was used.

5.4.1 Core Cluster Software

1. Register the server in RHN
2. Enable entitlements for the software channels:
   - RHEL Cluster-Storage
   - RHEL Clustering
3. Install the required RPMs with the following `yum` command:

   ```
   # yum install cman gfs-utils kmod-gfs lvm2-cluster rgmanager
   ```

5.4.2 Open-Sharedroot Software

Perform the following steps to install the open-sharedroot software packages:

1. Create the following yum configuration file */etc/yum.repos.d/comoonics.repo* for the open-sharedroot software channel:

   ```
   [comoonics]
   name=Packages for the comoonics shared root cluster
   baseurl=http://download.atix.de/yum/comoonics/redhat-el5/productive/noarch/
   enabled=1
   gpgcheck=1
   gpgkey=http://download.atix.de/yum/comoonics/comoonics-RPM-GPG.key

   [comoonics-$basearch]
   name=Packages for the comoonics shared root cluster
   baseurl=http://download.atix.de/yum/comoonics/redhat-el5/productive/$basearch
   enabled=1
   gpgcheck=1
   ```

```
gpgkey=http://download.atix.de/yum/comoonics/comoonics-RPM-GPG.key
```

Install the required open-sharedroot software packages:

```
# yum install comoonics-bootimage comoonics-cdsl-py comoonics-ec-py
  comoonics-cs-xsl-ec
```

5.5 Cluster Core Configuration

The cluster configuration file, */etc/cluster/cluster.conf* (in XML format), for this cluster contains has the following outline:

```
<?xml version="1.0"?>
<cluster>
    <cman/>
    <totem/>
    <quorumd/>
    <fence_daemon/>
    <clusternodes>
        <clusternode/>
        ...
        <clusternode/>
    </clusternodes>
    <fencedevices>
       <fencedevice>
       ...
       <fencedevice>
    </fencedevices>
    <rm>
        <!-- Configuration of the resource group manager -->
    </rm>
</cluster>
```

The cluster configuration can be created in three different ways:

1. **Conga**
   A web based configuration interface
2. **system-config-cluster**
   The local cluster configuration GUI
3. **vi**
   File editor

As some configuration details can only be defined using a common file editor such as *vi*, it is recommended to create the initial cluster configuration file with the help of a GUI based tool and later perform all necessary modifications by hand. To manually alter the configuration within a running cluster, the following steps must be performed:

1. Increment the *config_version* attribute within the <cluster> tag:

   ```
   <cluster config_version="X"/>
   ```

2. Update the ccs cluster configuration:

```
# ccs_tool update /etc/cluster/cluster.conf
```

The <cluster> tag should define the following attributes:

Attribute	Description
config_version	Version number of the configuration
Name	The name of the cluster

5.5.1 CMAN / OpenAIS

The OpenAIS daemon *aisexec* is started and configured by CMAN. Typically, all work is performed within the cman init script. The following command can be used to start the *aisexec* daemon:

```
# cman_tool join -w
```

Please consult the `cman_tool` manpage for detailed information.

The CMAN/OpenAIS portion within the cluster configuration file is defined within the <cman> tag. The following attributes should be taken into consideration:

Attribute	Description
expected_votes	Number of votes used to calculate the quorum
two_node	Special configuration option for 2-node clusters without a quorum disk (set to 1 to enable)

5.5.2 Qdisk

If the use a quorum device is intended, the following steps must be performed:

1. Format a shared disk partition as quorum disk:

```
# mkqdisk -c <device> - l <label>
```

2. Add a <quorumd> configuration tag to the cluster configuration file.
3. Optionally define helpful heuristics for qdiskd verification purposes

The <quorumd> tag should define the following attributes:

Attribute	Description
interval	The frequency of read/write cycles, in seconds.
tko	The number of cycles a node must miss in order to be declared dead.
votes	The number of votes the quorum daemon advertises

Attribute	Description
	to CMAN when it has a high enough score.
log_level	Controls the verbosity of the quorum daemon in the system logs. 0 = emergencies; 7 = debug.
log_facility	Controls the syslog facility used by the quorum daemon when logging. For a complete list of available facilities, see syslog.conf(5). The default value for this is 'daemon'.
min_score	Absolute minimum score to be consider one's self "alive". If omitted, or set to 0, the default function "floor((n+1)/2)" is used, where n is the total of all of defined heuristics' *score* attributes. This must never exceed the sum of the heuristic scores, or else the quorum disk will never be available.
device	The device the quorum daemon will use. This device must be the same on all nodes.
label	Overrides the device field if present. If specified, the quorum daemon will read /proc/partitions and search for qdisk signatures on every block device found, comparing the label against the specified label. This is useful in configurations where the block device name differs on a per-node basis.

In the case of a split brain situation, heuristics can be used to identify the cluster partition that is best to survive. Heuristics can be defined by adding a <heuristic> tag within the <quorumd> tag.

The <heuristic> tag should define the following attributes:

Attribute	Description
program	The program used to determine if this heuristic is alive. This can be anything executable by */bin/sh -c*. A return value of zero indicates success; anything else indicates failure.
score	The weight of this heuristic. Be careful when determining scores for heuristics. The default score for each heuristic is 1.
interval	The frequency (in seconds) at which we poll the heuristic. The default interval for every heuristic is 2 seconds.
tko	The number of heuristic test failures before a node is considered DOWN and its score is removed. The default tko for each heuristic is 1, which can be inadequate for actions such as 'ping'.

For more detailed information, refer to the qdisk man page.

If device mapper multipath is used together with qdiskd, the values for *tko* and *interval* must be carefully considered. In the example case of a path failover, all storage I/O will be queued by the device mapper module. The qdisk timeout must be adapted to the possible device mapper's queuing time.

5.5.3 Fencing

The fencing configuration consists of two parts. The first is the configuration of the fencing daemon (fenced) itself. The second is the configuration of the fencing agents that the daemon will use to fence each cluster node.

The fencing daemon is configured by adding the <fence_daemon> tag within the cluster configuration file. The following attributes should be considered:

Attribute	Description
post_join_delay	Post-join delay is the number of seconds the daemon will wait before fencing any victims after a node joins the domain.
post_fail_delay	Post-fail delay is the number of seconds the daemon will wait before fencing any victims after a domain member fails.
clean_start	Clean-start is used to prevent any startup fencing the daemon might do. It indicates that the daemon should assume all nodes are in a clean state to start.

The fencing agents used for each cluster node must be configured within the <fencedevices> tag. For each fence agent, a <fencedevice> tag within the <fencedevices> tag must be defined. The <fencedevice> tag should at minimum define the *agent* and *name* attributes. For further information about the different configuration options of all fencing agents, reference the man pages of the desired fencing agent.

5.5.4 Cluster Nodes

The configuration of the cluster nodes is controlled by <clusternode> tags within an encapsulating <clusternodes> tag.

The basic cluster node configuration should contain at least the following attributes:

Attribute	Description
name	The name of the host
nodeid	The id of the cluster node
votes	The number of quorum votes for this node

Within the <clusternode> tag, the methods used to fence the node must be defined. All

fencing mechanism are encapsulated within the <fence> tag. Each fencing mechanism is defined by the <method> tag.

Please refer to the man pages of fence as well as the man pages for the chosen fencing mechanisms for further details.

5.6 Local root Cluster Installation

For a cluster with a local root file system configuration, the following steps must be performed on **every** cluster node:

1. Install the Red Hat Enterprise Linux 5 operating system
2. Install the required cluster packages
3. Perform the required OS customizations
4. Perform the required network configurations
5. Perform the local storage configuration procedure
6. On **one** node only, create a cluster configuration file and copy the file to all other cluster members**.**
7. On **every** cluster node, start the cman init script:

   ```
   # service cman start
   ```

8. To verify the changes have been propagated, the version number and cluster status can be viewed on any node at any time using `cman_tool`.

   ```
   # cman_tool status
   ```

9. The state of all cluster nodes can be viewed with the following command:

   ```
   # cman_tool nodes
   ```

5.7 Shared Root Cluster Installation

Note that the shared root cluster installation is based on the open-sharedroot project. Please reference http://open-sharedroot.org/ for detailed information.

For a shared root file system configuration, the following steps must be performed:

On **one** node only:

1. Install the Red Hat Enterprise Linux 5 operating system
2. Install the required cluster packages
3. Perform the required OS customizations
4. Perform the required network configurations
5. Performed the required storage configurations
6. Create a cluster configuration file with shared root configuration settings
7. Install the required shared root software packages

After the installation of the first node is complete, the installation must be transferred to a shared root device and some modifications must be performed. For detailed installation steps, please reference the following documentation:

- NFS shared root

  http://open-sharedroot.org/documentation/rhel5-nfs-shared-root-mini-howto
- GFS shared root

  http://open-sharedroot.org/documentation/rhel5-gfs-shared-root-mini-howto
- Yum channel for shared root software packages

  http://open-sharedroot.org/faq/can-i-use-yum-or-up2date-to-install-the-software

After the shared root installation, all servers can be booted into the same root file system.

6 SAP Installation

The SAP installations described in this section include three different cluster configurations:

1. Local Root and Shared Storage with local File System Types
2. Local Root and Shared Storage with GFS
3. Shared Root and Shared Storage with GFS

6.1 Local Root and Shared Storage with Local File System Types

6.1.1 SAP Architecture

Following the established SAP documentation is highly recommended:

- SAP Installation Guide
  http://service.sap.com/instguides
- SAP Technical Infrastructure Guide
  https://www.sdn.sap.com/irj/sdn/ha

6.1.2 SAP Virtual IP Addresses

SAP NetWeaver is typically installed via the graphical installation tool *sapinst*. Before beginning the installation, determine which IP addresses and host names are preferred for use during the SAP installation. First, each node requires a static IP address and an associated host name. This address is also referred to as the physical IP address . Second, each database and SAP instance will require a virtual IP address / host name. The virtual addresses must not be configured at the operating system level because they are under the control of RHCS. Those addresses are referred to as the virtual IP addresses.

Local dialog instances, which are not part of the cluster, use a virtual host name as an alias to the physical host name so those SAP instances are not failed over by RHCS.

The enqueue replication instances do not need IP addresses because no connections are established with them. The virtual host name is only used to start the instances manually via the `sapstart` command and to distinguish their profile names from physical host names.

Edit the */etc/hosts* file on all nodes and add the virtual host names and their associated IP addresses. Additionally, add any other cluster relevant host name and address (e.g., the physical host names or addresses of the nodes) to */etc/hosts* so the DNS server is no longer a possible single point of failure.

6.1.3 SAP File Systems

The file systems for our scenario must be prepared before installing SAP NetWeaver.

File systems must be set up locally, on shared storage with local file system type (e.g., Ext3), and on a highly available NFS server.

6.1.3.1 Local File Systems

Directories such as */usr/sap* and */sapmnt* can be created locally.

Specific directories for SAP agents such as */usr/sap/ccms*, */usr/sap/<SID>/ccms* or */usr/sap/SMD* must be configured according to your SAP landscape.

The linking directory */usr/sap/<SID>/SYS* can reside locally because it contains only links to */sapmnt/<SID>*.

The directory */usr/sap/tmp* should also be locally.

6.1.3.2 Shared Storage File Systems

The instance directories */usr/sap/<SID>/<InstanceNo>* must be set up on a shared storage, so that these directories are able to perform a switchover triggered by the cluster software.

The database directories must be set up according to the SAP installation guide.

MaxDB Create */sapdb/programs/lib* and */sapdb/programs/runtime* locally on every node. See the post-processing section how to copy the files after the installation.

Follow the database file system configuration recommendations from the SAP installation guide. It is recommended to have physically different mount points for the program files and for saplog and sapdata.

Oracle

Create */oracle/client/10x_64/instantclient* locally on every node. See the post-processing section how to copy the binaries after the installation.

Follow the database file system configuration recommendations from the SAP installation guide. It is recommended to have different physical mount points for the program files and for origlog, mirrlog and sapdata.

NOTE: The configuration process gets more complex when multiple database instances of the same type run within the cluster. The program files must be accessible for every instance.

The mounts from shared storage must be added to the cluster configuration as file system resources to the failover service.

6.1.3.3 NFS Mounted File Systems

The */sapmnt/<SID>* file system should resist on a high available NFS to be available for additional application server outside the cluster.

The transport directory */usr/sap/trans* should also be exported via NFS according to the SAP landscape.

6.1.3.4 Before Starting the SAP Installation

Before installing SAP NetWeaver, mount all the file necessary systems. Be conscious of the overmount effect by mounting the hierarchically highest directories first.

6.1.4 Installation with *sapinst*

When starting the SAP installation tool *sapinst,* specify the virtual host name.

```
sapinst SAPINST_USE_HOSTNAME=<virtual hostname>
```

For each SAP and database instance installed, choose the installation option "*High-Availability System*" as described in the SAP installation guide.

6.1.5 Installation Post-Processing

6.1.5.1 Users, Groups and Home Directories

Create users and groups on the second node as they were created by the SAP installation on the first node. Use the same user and group IDs.

Depending on the Installation Master CD that was used for the SAP installation, the login profiles for the SAP administrator user (<sid>adm) and the database administrator user could differ. In older and non-HA installations, the user login profiles look similar to this one: *.sapenv_hostname.csh*

Using the host name in the user login profiles is a problem in an HA environment. By default, the profiles .login, .profile and .cshrc will search for two types of user login profiles: first for the one including the local host name (e.g., .dbenv_hostname.csh) and then for a version without the host name included. The latest versions of the InstMaster CDs will install both versions of the user login profiles. This could lead to some confusion for the administrator with regard to which version is used in which case. The removal of all user login profiles (that include a host name in the file name) is recommended. Do this for both the SAP administrator, <sid>adm, as well as the database administrator user.

6.1.5.2 Synchronizing Files and Directories

Copy */etc/services* or its values that were adjusted by *sapinst* (see SAP related entries at the end of the file) to all other nodes.

MaxDB

Copy the file */etc/opt/sdb* and the directory structure */usr/spool/sql* to the other nodes.

Make the directories (with their content) */sapdb/programs/lib* and */sapdb/programs/runtime* available even if the file system */sapdb* is not mounted. To do so, mount */sapdb* and copy the directories */sapdb/programs/lib* and */sapdb/programs/runtime* to a temporary directory. Unmount */sapdb* and copy them locally to */sapdb/programs/lib* and */sapdb/programs/runtime*. Do this on every node.

Oracle

Copy the files */etc/oratab* and */etc/oraInst.loc* to the other nodes.

Make the directory (with its content) /oracle/client/10x_64/instantclient available even if the file system /oracle is not mounted. To do so, mount /oracle and copy the directory */oracle/client/10x_64/instantclient* to a temporary directory. Unmount */oracle* and copy it locally to */oracle/client/10x_64/instantclient*. Do this on every node.

6.1.5.3 SAP Profiles

The most important SAP profile parameter for a clustered SAP system is *SAPLOCALHOST*. After the installation with *sapinst*, ensure that all SAP instance profiles contain this parameter. The value of the parameter must be the virtual host name specified during the installation.

As a general requirement, the SAP parameter *es/implementation* must be set to "std" in the SAP DEFAULT.PFL file. See SAP Note 941735. The SAPInstance resource agent cannot use the AUTOMATIC_RECOVERY function for systems that have this parameter set to "map".

In the START profiles, the parameter *SAPSYSTEM* must be set (default since 7.00).

6.1.5.4 SAP Release-specific Post-processing

For improved SAP hardware key determination in high-availability scenarios of SAP Note 1178686.

For SAP kernel release 4.6D, follow the instructions in appendix A1 of SAP Note 1008828.

For SAP kernel release 6.40, follow the instructions of SAP Note 877795.

For SAP kernel release 6.40, update the SAP kernel to at least patch level 208.

When using a SAP kernel 6.40, please read and implement the actions from the section "Manual post-processing" from SAP Note 995116.

6.1.5.5 Before Starting the Cluster

An empty work directory (*/usr/sap/<SID>/<Instance><Number>/work*) of an SAP instance results in a monitoring error of the SAPInstance resource agent. Every instance must be started manually in order for the correct entries to be written to the work directory. After a manual shutdown of the instances, the cluster is ready to control them.

Remember that the virtual IP addresses for the SAP instances you wish to start must be active. They can be started manually (e.g., with the Linux command `ip`) and then stopped again after shutting down the SAP instances.

6.1.6 Enqueue Replication Server

Follow the instructions of the official SAP Library:
http://help.sap.com/saphelp_nw2004s/helpdata/en/de/cf853f11ed0617e10000000a114084/frameset.htm

6.2 Local Root and Shared Storage with GFS

6.2.1 SAP Architecture

Following the established SAP documentation is highly recommended:

- SAP Installation Guide

  http://service.sap.com/instguides
- SAP Technical Infrastructure Guide

  https://www.sdn.sap.com/irj/sdn/ha

6.2.2 SAP Virtual IP Addresses

SAP NetWeaver is typically installed via the graphical installation tool *sapinst*. Before beginning the installation, determine which IP addresses and host names are preferred for use during the SAP installation. First, each node requires a static IP address and an associated host name. This address is also referred to as the physical IP address . Second, each database and SAP instance will require a virtual IP address / host name. The virtual addresses must not be configured at the operating system level because they are under the control of RHCS. Those addresses are referred to as the virtual IP addresses.

Local dialog instances, which are not part of the cluster, use a virtual host name as an alias to the physical host name so those SAP instances are not failed over by RHCS.

The enqueue replication instances do not need IP addresses because no connections are established with them. The virtual host name is only used to start the instances manually via the `sapstart` command and to distinguish their profile names from physical host names.

Edit the */etc/hosts* file on all nodes and add the virtual host names and their associated IP addresses. Additionally, add any other cluster relevant host name and address (e.g., the physical host names or addresses of the nodes) to */etc/hosts* so that the DNS server is no longer a single point of failure.

6.2.3 SAP File Systems

The file systems for our scenario must be prepared before installing SAP NetWeaver.

File systems must be set up locally, on shared storage with file system type GFS, and on a highly available NFS server.

6.2.3.1 Local File Systems

Directories such as */usr/sap* and */sapmnt* can be created locally.

The directory */usr/sap/tmp* should also be locally.

6.2.3.2 Shared Storage File Systems

By using GFS as file system for SAP and database instances, the directory */usr/sap/<SID>* and any lower directory can be a GFS mount point.

The database directories can also reside entirely on GFS. Follow the database file system setup recommendations from the SAP installation guide. It is recommended to have different physical mount points for specific directories.

The GFS mounts from shared storage must be added to */etc/fstab* so they get mounted at system boot.

6.2.3.3 NFS Mounted File Systems

The */sapmnt/<SID>* file system should resist on a high available NFS to be available for additional application server outside the cluster.

The transport directory */usr/sap/trans* should also be exported via NFS according to your SAP landscape.

6.2.3.4 Before Starting the SAP Installation

Before installing SAP NetWeaver, mount all the file necessary systems. Be conscious of the overmount-effect by mounting the hierarchically highest directories first.

6.2.4 Installation with *sapinst*

When starting the SAP installation tool *sapinst,* specify the virtual host name.

```
sapinst SAPINST_USE_HOSTNAME=<virtual hostname>
```

For each SAP and database instance installed, choose the installation option "*High-Availability System*" as it is described in the SAP installation guide.

6.2.5 Installation Post-Processing

6.2.5.1 Users, Groups and Home Directories

Create users and groups on the second node as they were created by the SAP installation on the first node. Use the same user ID and group ID.

Depending on the Installation Master CD that was used for the SAP installation, the login profiles for the SAP administrator user (<sid>adm) and the database administrator user might be different. In older and non-HA installations the user login profiles look similar to this one: *.sapenv_hostname.csh*

Using the host name in the user login profiles is a problem in a highly available environment. By default the profiles .login, .profile and .cshrc will search for two types of user login profiles: first for the one including the local host name (e.g., .dbenv_hostname.csh) and then for a version without the host name included. Latest versions of the InstMaster CDs will install both versions of the user login profiles. This might lead to some confusion for the administrator, regarding which version is used in which case. We recommend removing all user login profiles that include a host name in the file name. Do this for both users: the SAP administrator <sid>adm and the database administrator user.

6.2.5.2 Synchronizing Files and Directories

Copy the */etc/services* or its values that were adjusted by *sapinst* (see SAP related entries at the end of the file) to all nodes.

MaxDB
Copy the file */etc/opt/sdb* and the directory structure */usr/spool/sql* to the GFS mount point */sapdb* and create links to them.

Oracle

Copy the files */etc/oratab* and */etc/oraInst.loc* to the GFS mount point */oracle* and create links to them.

6.2.5.3 SAP Profiles

The most important SAP profile parameter for a clustered SAP system is *SAPLOCALHOST*. After the installation with *sapinst*, make sure that all SAP instance profiles contain this parameter. The value of the parameter must be the virtual host name specified during the installation.

As a general requirement the SAP parameter *es/implementation* must be set to “std” in the SAP DEFAULT.PFL file. See SAP Note 941735. The SAPInstance resource agent cannot use the AUTOMATIC_RECOVERY function for systems that have set this parameter to "map".

In the START profiles, the parameter *SAPSYSTEM* must be set (default since 7.00).

6.2.5.4 SAP Release-specific Post-processing

For improved SAP hardware key determination in high-availability scenarios, see SAP Note 1178686.

For SAP kernel release 4.6D, follow the instructions in appendix A1 of SAP Note 1008828.

For SAP kernel release 6.40, follow the instructions of SAP Note 877795.

For SAP kernel release 6.40, update the SAP kernel to patch level 208 or later.

When using SAP kernel 6.40, read and implement the actions from the *Manual post-processing* section in SAP Note 995116.

6.2.5.5 Before Starting the Cluster

An empty work directory (*/usr/sap/<SID>/<Instance><Number>/work*) of an SAP instance results in a monitoring error of the SAPInstance resource agent. Every instance must be started manually in order for the correct entries to be written to the work directory. After a manual shutdown of the instances, the cluster is ready to control them.

Remember that the virtual IP addresses for the SAP instances you wish to start must be active. They can be started manually (e.g., with the Linux command `ip`) and stopped again after shutting down the SAP instances.

6.2.6 Enqueue Replication Server

Follow the instructions of the official SAP Library:
http://help.sap.com/saphelp_nw2004s/helpdata/en/de/cf853f11ed0617e10000000a114084/frameset.htm

6.3 Shared Root and Shared Storage with GFS

6.3.1 SAP Architecture

Following the established SAP documentation is highly recommended:

- SAP Installation Guide

  http://service.sap.com/instguides

- SAP Technical Infrastructure Guide

  https://www.sdn.sap.com/irj/sdn/ha

6.3.2 SAP Virtual IP Addresses

SAP NetWeaver is typically installed via the graphical installation tool *sapinst*. Before beginning the installation, determine which IP addresses and host names are preferred for use during the SAP installation. First, each node requires a static IP address and an associated host name. This address is also referred to as the physical IP address . Second, each database and SAP instance will require a virtual IP address / host name. The virtual addresses must not be configured at the operating system level because they are under the control of RHCS. Those addresses are referred to as the virtual IP addresses.

Local dialog instances, which are not part of the cluster, use a virtual host name as an alias to the physical host name so those SAP instances are not failed over by RHCS.

The enqueue replication instances do not need IP addresses because no connections are established with them. The virtual host name is only used to start the instances manually via the `sapstart` command and to distinguish their profile names from physical host names.

Edit the */etc/hosts* file on all nodes and add the virtual host names and their associated IP addresses. Additionally, add any other cluster relevant host name and address (e.g., the physical host names or addresses of the nodes) to */etc/hosts* so that the DNS server is no longer a possible single point of failure.

6.3.3 SAP File Systems

The file systems for our scenario must be prepared before installing SAP NetWeaver.

File systems must be set up on shared root, on GFS shared storage, and on a highly available NFS server.

6.3.3.1 File Systems on Shared Root

File systems such as */usr/sap*, */sapmnt* and */home* can be created on shared root.

6.3.3.2 Shared Storage File Systems

By using GFS as file system for SAP and database instances, the directory */usr/sap/<SID>* and each deeper directory can be a GFS mount point.

The database directories can also completely reside on GFS. Follow the database file system setup recommendations from the SAP installation guide. It is recommended to have physically different mount points for specific directories.

The GFS mounts from shared storage must be added to */etc/fstab* so they get mounted at system boot.

6.3.3.3 NFS Mounted File Systems

The */sapmnt/<SID>* file system should resist on a high available NFS to be available to additional application server outside the cluster.

The transport directory */usr/sap/trans* should also be exported via NFS according to the SAP landscape.

6.3.3.4 Before Starting the SAP Installation

Before installing SAP NetWeaver, mount all the necessary file systems. Be conscious of the overmount-effect by mounting the hierarchically highest directories first.

6.3.4 Installation with *sapinst*

When starting the SAP installation tool *sapinst,* specify the virtual host name.

```
sapinst SAPINST_USE_HOSTNAME=<virtual hostname>
```

For each SAP and database instance installed, choose the installation option "*High-Availability System*" as it is described in the SAP installation guide.

6.3.5 Installation Post-Processing

6.3.5.1 Users, Groups and Home Directories

By using shared root, users and groups on the second node already exist.

Depending on the Installation Master CD that was used for the SAP installation, the login profiles for the SAP administrator user (<sid>adm) and the database administrator user might be different. In older and non-HA installations the user login profiles look similar to this one: *.sapenv_hostname.csh*

Using the host name in the user login profiles is a problem in a highly available environment. By default the profiles .login, .profile and .cshrc will search for two types of user login profiles: first for the one including the local host name (e.g., .dbenv_hostname.csh) and then for a version without the host name included. Later versions of the InstMaster CDs will install both versions of the user login profiles. This could lead to some confusion for the administrator, with regard to which version is used in which case. Removing all user login profiles (that include a host name in the file name) is recommended. Do this for both the SAP administrator, <*sid*>*adm*, as well as the database administrator user.

6.3.5.2 SAP Profiles

The most important SAP profile parameter for a clustered SAP system is *SAPLOCALHOST*. After the installation with *sapinst*, ensure that all SAP instance profiles contain this parameter. The value of the parameter must be the virtual host name specified during the installation.

As a general requirement the SAP parameter *es/implementation* must be set to “std” in the SAP DEFAULT.PFL file. See SAP Note 941735. The SAPInstance resource agent cannot use the AUTOMATIC_RECOVERY function for systems have set this parameter to "map".

In the START profiles, the parameter *SAPSYSTEM* must be set (default since 7.00).

6.3.5.3 SAP Release-specific Post-processing

For improved SAP hardware key determination in high-availability scenarios, see SAP Note 1178686.

For SAP kernel release 4.6D, follow the instructions in appendix A1 of SAP Note 1008828.

For SAP kernel release 6.40, follow the instructions of SAP Note 877795.

For SAP kernel release 6.40, update the SAP kernel to at least patch level 208.

When using a SAP kernel 6.40, please read and implement the actions from the section *Manual post-processing* section of SAP Note 995116.

6.3.5.4 Before Starting the Cluster

An empty work directory (*/usr/sap/<SID>/<Instance><Number>/work*) of an SAP instance results in a monitoring error of the SAPInstance resource agent. Every instance must be started manually in order for the correct entries to be written to the work directory. After a manual shutdown of the instances, the cluster is ready to control them.

Remember that the virtual IP addresses for the SAP instances you wish to start must be active. They can be started manually (e.g., with the Linux command `ip`) and stopped again after shutting down the SAP instances.

6.3.6 Enqueue Replication Server

Follow the instructions of the official SAP Library:
http://help.sap.com/saphelp_nw2004s/helpdata/en/de/cf853f11ed0617e10000000a114084/frameset.htm

7 Resource Manager

7.1 Cluster Resources

There are many types of configurable cluster resources. Reference the *Adding a Cluster Service to the Cluster* section of Configuring and Managing a Red Hat Cluster for more information.

The following resource types will be defined to provide the high availability functionality for SAP.

7.2 Configuration

The resource group manager is configured within the cluster configuration file */etc/cluster/cluster.conf*. The configuration is encapsulated within the <rm> tag. The resource manager configuration has the following basic layout:

```
<rm>
    <failoverdomains>
        <failoverdomain/>
        ...
        <failoverdomain/>
    </failoverdomains>
    <resources>
        <resource/>
        ...
        <resource/>
    </resources>
    <service/>
    ...
    <service>
    <events>
      <event/>
      [...]
    <events
</rm>
```

The following <rm> attributes can be defined:

Attribute	Description
log_level	The log level is number from 0..7, where 7 is 'debug' and 0 is 'emergency only'. The default value is 4.
log_facility	Log facility name, such as *daemon*, *local4*, or similar. The default value is daemon.
central_processing	The central_processing option is used to activate the event mechanism. Central_processing is needed to enable the hard and soft service dependencies and the follow_service dependency.

7.3 Failover Domains

The cluster can be divided into logical subsets of cluster nodes. A failover domain is an ordered subset of members to which a service can be bound.

Failover domains are configured in the cluster configuration file. The following outlines the

basic configuration schema:

```
<rm>
   <failoverdomains>
     <failoverdomain name="name" restricted="[0|1]" ordered="[0|1]"
nofailback="[0|1]">
       <failoverdomainnode name="node1" priority="1..100" />
       ...
     </failoverdomain>
     ...
  </failoverdomains>
</rm>
```

The failover domains can be configured in different ways. The following configuration attributes can be set to define the failover rules:

Attribute	Description
restricted domain	Services bound to the domain can only run on cluster members which are also members of the failover domain. If no members of the failover domain are available, the service is placed in the stopped state.
unrestricted domain	Services bound to this domain can run on all cluster members but will run on a member of the domain whenever one is available. This means that if a service is running outside of the domain and a member of the domain comes online, the service will migrate to that member.
ordered domain	The order specified in the configuration dictates the order of preference of members within the domain. The highest-ranking member of the domain will run the service whenever it is online. This means that if member A has a higher-rank than member B, the service will migrate to A if it was running on B if A transitions from offline to online.
unordered domain	Members of the domain have no order of preference; any member may run the service. Services will always migrate to members of their failover domain whenever possible, however, in an unordered domain.
nofailback	Enabling this option for an ordered failover domain will prevent automated fail-back after a more-preferred node rejoins the cluster.

7.4 Cluster Resources and Services

There are many types of cluster resources that can be configured. Resources are bundled together to highly available services; i.e., a service consists of one or more cluster resources.

Resources can be used by any cluster service that requires one. Once associated with a cluster service, it can be relocated by a cluster member if it deems it necessary, or manually through a GUI interface, a web interface (conga) or via command line. If any cluster member providing the service becomes unable to do so (e.g., due to hardware or software failure, network/connectivity loss, etc.), the service with all its resources will automatically migrate to an eligible member.

Reference the *Adding a Cluster Service to the Cluster* section of Configuring and Managing a Red Hat Cluster for more information.

Highly available cluster services are configured within the <service> tag. Consider defining the following attributes:

Attribute	Description	Required
name	The name of the service or resource group	Yes
domain	The failover domain associated with this service	No
autostart	If set to yes, the service will automatically be started after the cluster forms a quorum. If set to no, this resource group will start in the 'disabled' state after the cluster forms a quorum. Default is 1.	No
exclusive	If set, this resource group will only relocate to nodes which have no other resource groups running in the event of a failure.	No
recovery	This currently has three possible options: "restart" tries to restart failed parts of this resource group locally before attempting to relocate (default); "relocate" does not bother trying to restart the service locally; "disable" disables the resource group if any component fails. Note that any resource with a valid "recover" operation that can be recovered without a restart will be.	No

The following resource types will be defined to provide the high availability functionality for SAP.

7.4.1 IP

The *ip* resource defines an ipv4 or ipv6 network address. The following attributes can be defined:

Attribute	Description	Required
address	IPv4 or IPv6 address to use as a virtual IP resource.	Yes
monitor_link	Enabling this causes the status verification to fail if the link on the NIC to which this IP address is bound is not present.	No

7.4.2 Netfs

The *netfs* resource defines an NFS or CIFS mount. The following attributes can be defined:

Attribute	Description	Required
name	A symbolic name for the *netfs* resource	Only as reference
mountpoint	Path in file system hierarchy to mount this file system	Yes
host	The ip address or host name of the server hosting the network file system resource	Yes
export	NFS Export directory name or CIFS share	Yes
fstype	File System type (nfs, nfs4 or cifs)	No
force_unmount	If set, the cluster will kill all processes using this file system when the resource group is stopped. Otherwise, the unmount will fail, and the resource group will be restarted.	No
options	Provides a list of mount options. If none are specified, the NFS file system is mounted *-o sync*.	No
no_unmount	Do not unmount the file system during a stop or relocation operation	No

7.4.3 FS

The *fs* resource defines a standard local file system mount; i.e., a non clustered or otherwise shared file system. The following attributes can be defined:

Attribute	Description	Required
name	A symbolic name for the file system resource.	Only as

Attribute	Description	Required
		reference
mountpoint	Path within file system hierarchy at which to mount this file system.	Yes
device	Block device, file system label, or UUID of file system.	Yes
fstype	File system type. If not specified, mount(8) will attempt to determine the file system type.	No
force_unmount	If set, the cluster will kill all processes using this file system when the resource group is stopped. Otherwise, the unmount will fail, and the resource group will be restarted.	No
options	Provides a list of mount options. If none are specified, the NFS file system is mounted -o sync.	No
self_fence	If set and unmounting the file system fails, the node will immediately reboot. Generally, this is used in conjunction with force_unmount support, but it is not required.	No
force_fsck	If set, the file system will be verified (even if it is a journaled file system). This option is ignored for non-journaled file systems such as ext2.	No

7.4.4 SAPInstance

Within SAP instances there can be several services. Typically, one will find the defined services in the START profile of the related instance (Note: with SAP Release 7.10, the START profile content was moved to the instance profile). Not all of those services are worth monitoring by the cluster. For instance, failover of the SAP instance would not be preferred if the central syslog collector daemon failed.

Services monitored by the SAPInstance resource agent:

- disp+work
- msg_server
- enserver
- enrepserver
- jcontrol
- jstart

The reserve conclusion of this is that a SAP instance without any of these services will not work with the resource agent. One could think of a standalone gateway instance or a

standalone web dispatcher instance which will fail to work with the resource agent. The next version of the agent can have a parameter that could be used to select which services should be monitored. However, this does not mean that a SAP web dispatcher cannot be included in another SAP instance that uses one of the monitored services (e.g., a SCS instance running a *msg_server* and a *enserver*). In this case, the web dispatcher will be started and stopped (together with the other services) by the cluster. The web dispatcher is then not monitored, meaning a hung or dead *sapwebdisp* process will not cause a failover of the entire SAP instance. However, that may be exactly what is preferred.

All operations of the SAPInstance resource agent are performed by using the startup framework called SAP Management Console, or *sapstartsrv*, that was introduced with SAP kernel release 6.40. Reference additional information regarding the SAP Management Console in SAP Note 1014480.

Using this framework defines a clear interface for the cluster heartbeat and how it views the SAP system. The monitoring options for the SAP system are far superior than other methods such as monitoring the `ps` command for running processes or pinging the application. *sapstartsrv* uses SOAP messages to request the status of running SAP processes. As such, it can request status directly from the process itself, independent from other issues that can exist at the time.

sapstartsrv has four status states:

- GREEN = everything is fine
- YELLOW = something is wrong, but the service is still working
- RED = the service does not work
- GRAY = the service has not been started

The SAPInstance resource agent will interpret GREEN and YELLOW as acceptable, meaning minor problems will not be reported to the heartbeat cluster. This prevents the cluster from performing an unwanted failover. The statuses RED and GRAY are reported as NOT_RUNNING to the cluster. Depending on the status the cluster expects from the resource, it will perform a restart, a failover, or do nothing.

Attribute	Description	Required
InstanceName	The full qualified SAP instance name. In the format: SID_INSTANCE_VHOST e.g., RHC_DVEBMGS01_lsrhcaci This is typically the name of the SAP instance profile. DEFAULT: none	Yes
DIR_EXECUTABLE	The fully qualified path to sapstartsrv and sapcontrol. Specify this parameter if the SAP kernel directory location was been changed	No

Attribute	Description	Required
	after the default SAP installation. DEFAULT: */usr/sap/<SID>/<INSTANCE>/exe* *or /usr/sap/<SID>/SYS/exe/run*	
DIR_PROFILE	The fully qualified path to the SAP START profile. Specify this parameter, if you have changed the SAP profile directory location after the default SAP installation. DEFAULT: */usr/sap/<SID>/SYS/profile*	No
START_PROFILE	The name of the SAP START profile. Specify this parameter if the name of the SAP START profile was changed after the SAP installation. The Instance Profile must be specified because SAP release 7.10 no longer has a START profile. DEFAULT: *START_<INSTANCE>_<VHOST>*	No (Yes for SAP 7.10)
START_WAITTIME	The time in seconds before a monitor operation is executed by the resource agent. If the monitor returns SUCCESS, the start is handled as SUCCESS. This is useful for resolving timing issues with the J2EE-AddIn instance. Typically, the resource agent waits until all services are started and the SAP Management Console reports a GREEN status. A double stack installation (ABAP + Java AddIn) consists of an ABAP dispatcher and a JAVA instance. Normally, the start of the JAVA instance takes longer than the start of the ABAP instance. For a JAVA Instance, one may need to configure a much higher timeout for the start operation of the resource in Heartbeat. The disadvantage would be that the discovery of a failed start by the cluster will take longer. For some, it may	No

Attribute	Description	Required
	be more important that the ABAP instance be up and running. A failure of the JAVA instance will not cause a failover of the SAP instance. Actually, the SAP MC reports a YELLOW status if the JAVA instance of a double stack system fails. From the perspective of the resource agent, a YELLOW status indicates all is well. Setting START_WAITTIME to a lower value causes the resource agent to verify the status of the instance during a start operation after that time. Where it would normally wait for a GREEN status, it now reports SUCCESS in the case of a YELLOW status after the specified time. This is only useful for double stack systems. DEFAULT: 3600	
AUTOMATIC_ RECOVER	The SAPInstance resource agent attempts to recover a failed start attempt automatically one time. This is accomplished by killing any running instance processes and executing cleanipc. Sometimes a crashed SAP instance leaves some processes and/or shared memory segments behind. Setting this option to true will try to remove those leftovers during a start operation, reducing administrator labor. DEFAULT: false	No
PRE_START_ USEREXIT POST_START_ USEREXIT PRE_STOP_ USEREXIT	The fully qualified path to a script or program which should be executed before/after a resource is started/stopped. SAP systems often required additional software run on the same server. That can be monitoring software or software for some interfaces the SAP system	No

Attribute	Description	Required
POST_STOP_ USEREXIT	uses. Those programs can include by writing an OCF resource agent into the Heartbeat cluster. However, sometimes writing a resource agent is too much effort for this task. With the provided userexits, one can easily include their own scripts, that do not follow the OCF standard, into the cluster. Note that the returncode of said script will not be used by the SAPInstance resource agent. The call of userexit is syncron, meaning the time the script requires is going into the timeout of the start/stop operation defined in the Heartbeat cluster configuration. If the script hangs, SAP may not be started. DEFAULT: empty	

7.4.5 SAPDatabase

The purpose of the resource agent is to start, stop, and monitor the database instance of an SAP system. Together with the RDBMS system, it will also control the related network service for the database (such as the Oracle Listener or the MaxDB xserver). The resource agent expects a standard SAP installation and therefore requires less parameters configured.

The monitor operation of the resource agent can test the availability of the database by using SAP tools (*R3trans* or *jdbcconnect*). With that, it ensures that the database is truly accessible by the SAP system.

After an unclean exit or crash of a database, require a recover procedure to restart can be required. The resource agent has a procedure implemented for each database type. If preferred, the attribute AUTOMATIC_RECOVER provides this functionality.

Attribute	Description	Required
SID	The unique SAP system identifier. e.g., RHC DEFAULT: empty	Yes
DBTYPE	The name of the database vendor used. Set either: ORA,DB6,ADA DEFAULT: empty	Yes

Attribute	Description	Required
DIR_EXECUTABLE	The full qualified path to the SAP kernel. The resource agent requires the *startdb* and the *R3trans* executables. For that reason, the directory with the SAP kernel must be accessible to the database server at any given time. Specify this parameter if the SAP kernel directory location was changed after the default SAP installation. DEFAULT: */usr/sap/<SID>/<INSTANCE>/exe* or */usr/sap/<SID>/SYS/exe/run* or */sapmnt/<SID>/exe*	No
NETSERVICENAME	The Oracle TNS listener name DEFAULT: LISTENER (if DBTYPE = ORA)	No
DBJ2EE_ONLY	If no ABAP stack is installed in the SAP database, set this to true. Non ABAP systems cannot be monitored using *R3trans*. That parameter shifts the monitoring method to *jdbcconnect*. DEFAULT: false	No
JAVA_HOME	This is required only if the DBJ2EE_ONLY parameter is set to true. Enter the path to the Java SDK used by the SAP WebAS Java. Set this parameter if the environment variable JAVA_HOME is not set for the root user, or points to another directory than that of the JAVA_HOME for the sidadm user. DEFAULT: $JAVA_HOME	No
STRICT_ MONITORING	Controls how the resource agent monitors the database. If true, it will use SAP tools to test the connection to the database.	No

Attribute	Description	Required
	Not fort use with Oracle as it will result in unwanted failovers in the case of a stuck archiver. DEFAULT: false	
AUTOMATIC_ RECOVER	The SAPDatabase resource agent tries to recover a failed start attempt automatically one time. This is achieved by performing a forced abort of the RDBMS and/or executing recovery commands. DEFAULT: false	No
DIR_BOOTSTRAP	The full qualified path to the J2EE instance bootstrap directory. e.g., */usr/sap/RHC/J00/j2ee/cluster/bootstrap* This is required only if the DBJ2EE_ONLY parameter is set to true. Specify this parameter the SAP j2ee bootstrap directory location was changed after the default SAP installation. DEFAULT: */usr/sap/<SID>/*/j2ee/cluster/bootstrap*	No
DIR_SECSTORE	The full qualified path to the J2EE security store directory. This is required only if the DBJ2EE_ONLY parameter is set to true. Specify this parameter if the SAP j2ee secure store directory location was changed after the default SAP installation. DEFAULT: */usr/sap/<SID>/SYS/global/security/lib/tools*	No
DB_JARS	The full qualified file name of the jdbc driver for the database connection test.	No (Yes for SAP 7.10)

Attribute	Description	Required
	This is required only if the DBJ2EE_ONLY parameter is set to true. It will be automatically read from the bootstrap.properties file in Java engine 6.40 and 7.00. For Java engine 7.10, the parameter is mandatory. Example: */oracle/client/10x_64/instantclient/libclntsh.so* DEFAULT: empty	
PRE_START_USEREXIT POST_START_USEREXIT PRE_STOP_USEREXIT POST_STOP_USEREXIT	The full qualified path to a script or program that should be executed before/after this resource was started/stopped. SAP systems often required additional software to run on the same server. That can be monitoring software or software for some interfaces the SAP system uses. Those programs can be included by writing their own OCF resource agent into the Heartbeat cluster. However, sometimes writing a resource agent is too much effort for this task. With the provided userexits, one can easily include their own scripts, that do not follow the OCF standard, into the cluster. Note that the returncode of such a script will not be used by the SAPInstance resource agent. The call of userexit is syncron, meaning the time the script requires is going into the timeout of the start/stop operation defined in the Heartbeat cluster configuration. If the script hangs, the database may not be started. DEFAULT: empty	No

7.5 Dependencies

7.5.1 Resource Dependencies

The resources within a cluster service follow two different dependency rules.

First, the nesting within the service configuration defines startup order and resource dependencies.

In the following example, resource2 depends on resource1. In addition, resource1 is started prior to starting resource2.

```
<service>
    <resource name="resource1">
        <resource name="resource2">
        </resource>
    </resource>
</service>
```

Second, an implicit order and dependency is defined. The following lists the implicit startup order of the previously defined resources:

1. fs
2. netfs
3. ip
4. other

7.5.2 Service Dependencies

The services in a cluster sometimes require some dependency rules; e.g., sometimes a database must be started prior to starting the application server. However, if the database fails and is subsequently relocated, the application server should survive. Additionally, SAP enqueue replication requires a special type of dependency. The enqueue server must always follow the replicated enqueue.

7.5.2.1 Hard and Soft Dependencies

The rgmanager service defines inter-service dependencies with soft and hard requirements. A hard dependency would cause the dependent service to be stopped/started if its dependency were stopped/started.

A soft dependency is only valid for initial startup. The dependent service would not be stopped if its dependency were stopped.

The following example defines the service configuration for a soft dependency:

```
<service name="service1"/>
<service name="service2" depend="service:service1" depend_mode="soft"/>
```

Note that *central_processing* must be enabled to enable hard and soft service dependencies.

7.5.2.2 Follow Service Dependency

The follow service dependency makes use of rgmanager's RIND event scripting mechanism. In order to activate the follow service dependency, *central_processing* must be enabled. Also, the following events must be defined within the <rm> tag:

```
<rm>
  <events>
    <event class="service" name="service-ers">
      notice("Event service triggered!");
      evalfile("/usr/share/cluster/follow-service.sl");
      follow_service("service:svc1", "service:svc2", "service:mastersvc");
    </event>
    <event class="node" name="node-ers">
      notice("Event node triggered!");
      evalfile("/usr/share/cluster/follow-service.sl");
      follow_service("service:svc1", "service:svc2", "service:mastersvc");
    </event>
  </events>
</rm>
```

The event configuration for follow_service requires the following service names to be defined:

Service	Description
svc1	The name of the service that should follow *svc2* during a failover operation.
svc2	The name of the service that should be followed by *svc1* during a failover operation.
mastersvc	The name of the service that should be started if only one node is available.

During a failover operation, the follow service mechanism is working with the following rules:

Case 1: svc1 has to be relocated:

svc1 is relocated to a server where svc2 is not running.

If only one server is available and svc2 is running on it, svc1 will be relocated to the node only if mastersvc==svc1,

Case 2: svc2 has to be relocated:

svc2 is relocated to the server where svc1 is running.

svc1 is relocated to another server.

If only one server is available, the service mastersvc==svc1|svc2 will be started on this node.

8 Cluster Management

8.1 CMAN

The basic cluster operation can be verified using the `cman_tool` utility

8.1.1 cman_tool status

The `cman_tool status` command can be used to show the status of one cluster node:

```
# cman_tool status
Version: 6.1.0
Config Version: 59
Cluster Name: lsrhc5
Cluster Id: 6875
Cluster Member: Yes
Cluster Generation: 24
Membership state: Cluster-Member
Nodes: 2
Expected votes: 3
Quorum device votes: 1
Total votes: 3
Quorum: 2
Active subsystems: 10
Flags: Dirty
Ports Bound: 0 11 177
Node name: ls3198a
Node ID: 1
Multicast addresses: 239.192.26.245
Node addresses: 192.168.10.1
```

8.1.2 cman_tool nodes

The `cman_tool nodes` command can be used to show the status of the basic cluster configuration:

```
# cman_tool nodes
Node  Sts   Inc   Joined               Name
   0   M      0   2009-02-19 21:06:33  /dev/disk/by-id/scsi-
360060160eda508008e55ad9b6a54db11-part1
   1   M      4   2009-02-19 21:05:35  ls3198a
   2   M     24   2009-02-20 15:02:37  ls3199a
```

8.1.3 cman_tool services

The `cman_tool services` command can be used to display the status of the core cluster services:

```
# cman_tool services
type             level name        id       state
fence            0     default     00010001 none
[1 2]
```

```
dlm             1     root        00040001 none
[1 2]
dlm             1     clvmd       00020001 none
[1]
dlm             1     rhc_ascs    00020002 none
[1 2]
dlm             1     rhc_usrsap  00040002 none
[1 2]
dlm             1     rhc_ers     00060002 none
[1 2]
dlm             1     rhc_oracle  00080002 none
[1 2]
dlm             1     rgmanager   00090002 none
[1 2]
gfs             2     root        00030001 none
[1 2]
gfs             2     rhc_ascs    00010002 none
[1 2]
gfs             2     rhc_usrsap  00030002 none
[1 2]
gfs             2     rhc_ers     00050002 none
[1 2]
gfs             2     rhc_oracle  00070002 none
[1 2]
```

8.2 rgmanager

8.2.1 clustat

The resource group or service manager state can be verified with the `clustat` command:

```
# clustat
Cluster Status for lsrhc5 @ Fri Feb 20 15:13:52 2009
Member Status: Quorate

 Member Name                                ID   Status
 ------ ----                                ---- ------
 ls3198a                                      1 Online, Local, RG-Master
 ls3199a                                      2 Online, RG-Worker
 /dev/disk/by-id/scsi-360060160eda508008e     0 Online

 Service Name                  Owner (Last)                  State
 ------- ----                  ----- ------                  -----
 service:rhc_aci               ls3198a                       started
 service:rhc_ascs              ls3198a                       started
 service:rhc_ers               ls3199a                       started
 service:rhc_oradb             ls3198a                       started
```

8.2.2 clusvcadm

The resource manager services can be controlled by the `clusvcadm` command. The basic operations are:

- `clusvcadm -e <service> -m <member>`

  starts service <service> on member <member>
- `clusvcadm -r <service> -m <member>`

  relocate service <service> to member <member>
- `clusvcadm -d <service>`

  disables/stops service <service>

For detailed information, reference the `clusvcadm(8)` manpage.

8.2.3 rg_test

The resource manager cluster configuration can be verified using the `rg_test` command:

```
# rg_test test /etc/cluster/cluster.conf
Running in test mode.
Loaded 22 resource rules
=== Resources List ===
Resource type: ip
Instances: 1/1
Agent: ip.sh
Attributes:
[...]
```

8.3 Open-Sharedroot

In the following section, some important tips for the open-sharedroot cluster administration are given.

Please reference the Open-Sharedroot Cluster Administration Handbook for more information.

8.3.1 Host dependent files

In a sharedroot cluster, some files need to be made host dependent. The `com-mkcdsl` command is used to manage host dependent files and directories.

The basic operations are:

- `com-mkcdsl -a <file>`

  will make the file or directory <file> host dependent.
- `com-mkcdsl -s <file>`

  will make the file or directory <file> shared again.

Further information can be gained from the `com-mkcdsl` man page.

8.3.2 Update initrd

In a sharedroot cluster, the cluster configuration file */etc/cluster/cluster.conf* must be copied into the initrd. Therefore the process of updating the cluster configuration is combined with updating the initrd.

1. Copy the file */opt/atix/comoonics-cs/xsl/updateinitrd.xml to* /etc/comonics/enterprisecopy/updateinitrd.xml
2. Adjust the settings of /etc/comonics/enterprisecopy/updateinitrd.xml to fit your needs, e.g., define the correct boot device:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE enterprisecopy SYSTEM "/opt/atix/comoonics-cs/xml/comoonics-
enterprise-copy.dtd">
<enterprisecopy>
  <modificationset type="filesystem">
    <device name="/dev/mapper/DGC_010p1">
    [...]
```

3. Use the comoonics enterprise copy tool `com-ec` to begin the update process. Do not forget to increment the *cluster_version* value in advance.

```
# com-ec /etc/comonics/enterprisecopy/updateinitrd.xml
```

8.3.3 Cluster Clone

The comoonics enterprise copy software solution can be used to create a bootable or archived clone of the diskless sharedroot cluster. To create a bootable clone, the following steps must be performed:

1. Copy the file */opt/atix/comoonics-cs/xsl/localclone.xsl* to */etc/comoonics/enterprisecopy/localclone.xml*
2. Adjust the settings of */etc/comoonics/enterprisecopy/localclone.xml* to fit your needs. Ensure that the source disks and destination disks are correct. Be aware that all data on the target destination disk will be deleted during a cloning process.:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE localclone SYSTEM "/opt/atix/comoonics-cs/xml/comoonics-
enterprise-clone.dtd">
<localclone name="localclone2disk" source="disk" destination="disk">
  <cluster name="lsrhc5" sourcesuffix="" destsuffix="C"/>
  <sourcedisks>
     <bootdisk name="/dev/mapper/DGC_010"/>
     <rootdisk name="/dev/mapper/DGC_000"/>
  </sourcedisks>
  <destdisks>
    <bootdisk name="/dev/mapper/DGC_009"/>
    <rootdisk name="/dev/mapper/DGC_004"/>
  </destdisks>
  <kernel version="2.6.18-128.el5"/>
</localclone>
```

3. Execute the cloning process with the following command:

```
# com-ec -x /opt/atix/comoonics-cs/xsl/localclone.xsl /etc/comoonics/
```

```
enterprisecopy/localclone.xml
```

For detailed information regarding the comoonics enterprise software solution, please consult the *Comoonics Enterprise Copy* section of the Open-Sharedroot Administrators Handbook.

Appendix A: cluster.conf

The following */etc/cluster/cluster.conf* file content was used during testing.

```
<?xml version="1.0"?>
<cluster config_version="60" name="lsrhc5">
  <cman expected_votes="3" two_node="0"/>
  <totem token="41000"/>
  <quorumd interval="2" tko="10" votes="1" label="quorum4">
  </quorumd>
  <fence_daemon clean_start="1" post_fail_delay="0" post_join_delay="3"/>
  <clusternodes>
    <clusternode name="ls3198a" votes="1" nodeid="1">
      <com_info>
        <scsi failover="mapper"/>
        <syslog name="ls3199a"/>
        <rootvolume name="/dev/vg_lsrhc5_sr/lv_sharedroot"/>
        <eth name="eth1" mac="00:17:31:A7:83:7F" ip="192.168.10.1"
mask="255.255.255.0" gateway=""/>
        <fenceackserver user="root" passwd="xxx"/>
      </com_info>
      <fence>
        <method name="1">
          <device name="ipmi" ipaddr="10.20.89.3"/>
        </method>
        <method name="2">
          <device name="fence_manual" nodename="ls3198a"/>
        </method>
      </fence>
    </clusternode>
    <clusternode name="ls3199a" votes="1" nodeid="2">
      <com_info>
        <scsi failover="mapper"/>
        <syslog name="ls3198a"/>
        <rootvolume name="/dev/vg_lsrhc5_sr/lv_sharedroot"/>
        <eth name="eth1" mac="00:17:31:A7:83:E1" ip="192.168.10.2"
mask="255.255.255.0" gateway=""/>
        <fenceackserver user="root" passwd="sap"/>
      </com_info>
      <fence>
        <method name="1">
          <device name="ipmi" ipaddr="10.20.89.4"/>
        </method>
        <method name="2">
          <device name="fence_manual" nodename="ls3199a"/>
        </method>
      </fence>
    </clusternode>
```

```
  </clusternodes>
  <fencedevices>
    <fencedevice name="ipmi" agent="fence_ipmilan" auth="password"
login="admin" passwd="XXX" option="reboot"/>
    <fencedevice agent="fence_manual" name="fence_manual"/>
  </fencedevices>
  <rm log_level="7" central_processing="1">
    <failoverdomains>
      <failoverdomain name="ALL">
        <failoverdomainnode name="ls3199a"/>
        <failoverdomainnode name="ls3198a"/>
      </failoverdomain>
    </failoverdomains>
    <resources>
      <ip address="10.20.88.232" monitor_link="1"/>
      <ip address="10.20.88.233" monitor_link="1"/>
      <ip address="10.20.88.234" monitor_link="1"/>
      <ip address="10.20.88.235" monitor_link="1"/>
      <ip address="10.20.88.236" monitor_link="1"/>
      <ip address="10.20.88.237" monitor_link="1"/>
      <ip address="10.20.88.238" monitor_link="1"/>
      <ip address="10.20.88.239" monitor_link="1"/>
      <netfs name="fssapmnt" no_umount="yes" host="ls3197v3"
export="/export/sapmnt/RHC" mountpoint="/sapmnt/RHC"/>
    </resources>
            <service autostart="0" exclusive="0" name="rhc_oradb"
recovery="relocate" domain="ALL">
      <netfs ref="fssapmnt"/>
      <ip ref="10.20.88.232"/>
      <SAPDatabase SID="RHC" DBTYPE="ORA"/>
    </service>
             <service autostart="0" exclusive="0" name="rhc_ascs"
recovery="relocate" domain="ALL" depend="service:rhc_oradb"
depend_mode="soft">
      <netfs ref="fssapmnt"/>
      <ip ref="10.20.88.235"/>
      <SAPInstance InstanceName="RHC_ASCS00_lsrhcascs"/>
    </service>
    <service autostart="0" exclusive="0" name="rhc_aci" recovery="relocate"
domain="ALL" depend="service:rhc_ascs" depend_mode="soft">
      <netfs ref="fssapmnt"/>
      <ip ref="10.20.88.233"/>
      <SAPInstance InstanceName="RHC_DVEBMGS01_lsrhcaci"
AUTOMATIC_RECOVER="true"/>
    </service>
    <service autostart="0" exclusive="0" name="rhc_ers" recovery="relocate"
domain="ALL">
      <netfs ref="fssapmnt"/>
      <ip ref="10.20.88.236"/>
      <SAPInstance InstanceName="RHC_ERS10_lsrhcaenr"/>
    </service>
    <events>
      <event class="service" name="service-ers">
        notice("Event service triggered!");
```

```
        evalfile("/usr/local/cluster/follow-service.sl");
        follow_service("service:rhc_ascs", "service:rhc_ers",
"service:rhc_ascs");
      </event>
      <event class="node" name="node-ers">
        notice("Event node triggered!");
        evalfile("/usr/local/cluster/follow-service.sl");
        follow_service("service:rhc_ascs", "service:rhc_ers",
"service:rhc_ascs");
      </event>
    </events>
  </rm>
</cluster>
```

Appendix B: multipath.conf

The following */etc/multipath.conf* file content was used during testing.

```
## This is the /etc/multipath.conf file recommended for
## EMC storage devices.
##
## OS : RHEL 4 U3
## Arrays : CLARiiON and Symmetrix
##
## Use user friendly names, instead of using WWIDs as names.
defaults {
        ## Use user friendly names, instead of using WWIDs as names.
        user_friendly_names yes
}

## The blacklist is the enumeration of all devices that are to be
## excluded from multipath control

blacklist {
        ## Replace the wwid with the output of the command
        ## 'scsi_id -g -u -s /block/[internal scsi disk name]'
        ## Enumerate the wwid for all internal scsi disks.
        ## Optionally, the wwid of VCM database may also be listed here.
        ##
        #wwid 35005076718 d4224d
        #wwid 3600a0b80002610520000412456baf9f
        device {
          vendor "ATA"
          product ".*"
        }
        devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
        devnode "^hd[a-z][0-9]*"
        devnode "^cciss!c[0-9]d[0-9]*"
}

devices {
        ## Device attributes requirements for EMC Symmetrix
        ## are part of the default definitions and do not require separate
        ## definition.
```

```
        ## Device attributes for EMC CLARiiON
        device {
          vendor "DGC "
          product "*"
          getuid_callout "/sbin/scsi_id -g -u -s /block/%n"
          failback manual
        }
#
multipaths {
multipath {
   wwid 360060160eda508008c55ad9b6a54db11
   alias DGC_000
}

multipath {
   wwid 360060160eda508008d55ad9b6a54db11
   alias DGC_001
}
multipath {
   wwid 360060160eda508003cb2f077e8f1db11
   alias DGC_002
}
multipath {
   wwid 360060160eda508008e55ad9b6a54db11
   alias DGC_003
}
multipath {
   wwid 360060160eda508008f55ad9b6a54db11
   alias DGC_004
}
multipath {
   wwid 360060160eda508009c55ad9b6a54db11
   alias DGC_005
}
multipath {
   wwid 360060160eda508009e55ad9b6a54db11
   alias DGC_006
}
multipath {
   wwid 360060160eda50800a055ad9b6a54db11
   alias DGC_007
}
multipath {
   wwid 360060160eda50800a355ad9b6a54db11
   alias DGC_009
}
multipath {
   wwid 360060160eda5080043b2f077e8f1db11
   alias DGC_010
}
multipath {
   wwid 360060160eda50800a255ad9b6a54db11
   alias DGC_011
}
```

```
multipath {
   wwid 360060160eda50800a155ad9b6a54db11
   alias DGC_012
}
multipath {
   wwid 360060160eda508009f55ad9b6a54db11
   alias DGC_013
}
multipath {
   wwid 360060160eda508009d55ad9b6a54db11
   alias DGC_014
}
multipath {
   wwid 360060160eda508009055ad9b6a54db11
   alias DGC_015
}
multipath {
   wwid 360060160eda508009155ad9b6a54db11
   alias DGC_016
}
multipath {
   wwid 360060160eda508009255ad9b6a54db11
   alias DGC_017
}
multipath {
   wwid 360060160eda508009355ad9b6a54db11
   alias DGC_018
}
```

Appendix C: lvm.conf

The following */etc/lvm/lvm.conf* file content was used during testing.

```
# This is an example configuration file for the LVM2 system.
# It contains the default settings that would be used if there was no
# /etc/lvm/lvm.conf file.
#
# Refer to 'man lvm.conf' for further information including the file
layout.
#
# To put this file in a different directory and override /etc/lvm set
# the environment variable LVM_SYSTEM_DIR before running the tools.


# This section allows you to configure which block devices should
# be used by the LVM system.
devices {

    # Where do you want your volume groups to appear ?
    dir = "/dev"

    # An array of directories that contain the device nodes you wish
    # to use with LVM2.
    #scan = [ "/dev" ]
```

```
    scan = [ "/dev/mapper" ]

    # If several entries in the scanned directories correspond to the
    # same block device and the tools need to display a name for device,
    # all the pathnames are matched against each item in the following
    # list of regular expressions in turn and the first match is used.
    preferred_names = [ ]

    # preferred_names = [ "^/dev/mpath/", "^/dev/[hs]d" ]

    # A filter that tells LVM2 to only use a restricted set of devices.
    # The filter consists of an array of regular expressions.  These
    # expressions can be delimited by a character of your choice, and
    # prefixed with either an 'a' (for accept) or 'r' (for reject).
    # The first expression found to match a device name determines if
    # the device will be accepted or rejected (ignored).  Devices that
    # don't match any patterns are accepted.

    # Be careful if there there are symbolic links or multiple filesystem
    # entries for the same device as each name is verifyed separately
against
    # the list of patterns.  The effect is that if any name matches any 'a'
    # pattern, the device is accepted; otherwise if any name matches any
'r'
    # pattern it is rejected; otherwise it is accepted.

    # Don't have more than one filter line active at once: only one gets
used.

    # Run vgscan after you change this parameter to ensure that
    # the cache file gets regenerated (see below).
    # If it doesn't do what you expect, verify the output of 'vgscan
-vvvv'.


    # By default we accept every block device:
    filter = [ "a/.*/" ]

    # Exclude the cdrom drive
    # filter = [ "r|/dev/cdrom|" ]

    # When testing I like to work with just loopback devices:
    # filter = [ "a/loop/", "r/.*/" ]

    # Or maybe all loops and IDE drives except hdc:
    # filter =[ "a|loop|", "r|/dev/hdc|", "a|/dev/ide|", "r|.*|" ]

    # Use anchors if you want to be really specific
    # filter = [ "a|^/dev/hda8$|", "r/.*/" ]

    # The results of the filtering are cached on disk to avoid
    # rescanning dud devices (which can take a very long time).
    # By default this cache is stored in the /etc/lvm/cache directory
    # in a file called '.cache'.
```

```
    # It is safe to delete the contents: the tools regenerate it.
    # (The old setting 'cache' is still respected if neither of
    # these new ones is present.)
    cache_dir = "/etc/lvm/cache"
    cache_file_prefix = ""

    # You can turn off writing this cache file by setting this to 0.
    write_cache_state = 1

    # Advanced settings.

    # List of pairs of additional acceptable block device types found
    # in /proc/devices with maximum (non-zero) number of partitions.
    # types = [ "fd", 16 ]

    # If sysfs is mounted (2.6 kernels), restrict device scanning to
    # the block devices it believes are valid.
    # 1 enables; 0 disables.
    sysfs_scan = 1

    # By default, LVM2 will ignore devices used as components of
    # software RAID (md) devices by looking for md superblocks.
    # 1 enables; 0 disables.
    md_component_detection = 1

    # If, while scanning the system for PVs, LVM2 encounters a device-
mapper
    # device that has its I/O suspended, it waits for it to become
accessible.
    # Set this to 1 to skip such devices.  This should only be needed
    # in recovery situations.
    ignore_suspended_devices = 0
}

# This section that allows you to configure the nature of the
# information that LVM2 reports.
log {

    # Controls the messages sent to stdout or stderr.
    # There are three levels of verbosity, 3 being the most verbose.
    verbose = 0

    # Should we send log messages through syslog?
    # 1 is yes; 0 is no.
    syslog = 1

    # Should we log error and debug messages to a file?
    # By default there is no log file.
    #file = "/var/log/lvm2.log"

    # Should we overwrite the log file each time the program is run?
    # By default we append.
    overwrite = 0
```

```
    # What level of log messages should we send to the log file and/or
syslog?
    # There are 6 syslog-like log levels currently in use - 2 to 7
inclusive.
    # 7 is the most verbose (LOG_DEBUG).
    level = 0

    # Format of output messages
    # Whether or not (1 or 0) to indent messages according to their
severity
    indent = 1

    # Whether or not (1 or 0) to display the command name on each line
output
    command_names = 0

    # A prefix to use before the message text (but after the command name,
    # if selected).  Default is two spaces, so you can see/grep the
severity
    # of each message.
    prefix = "  "

    # To make the messages look similar to the original LVM tools use:
    #   indent = 0
    #   command_names = 1
    #   prefix = " -- "

    # Set this if you want log messages during activation.
    # Don't use this in low memory situations (can deadlock).
    # activation = 0
}

# Configuration of metadata backups and archiving.  In LVM2 when we
# talk about a 'backup' we mean making a copy of the metadata for the
# *current* system.  The 'archive' contains old metadata configurations.
# Backups are stored in a human readable text format.
backup {

    # Should we maintain a backup of the current metadata configuration ?
    # Use 1 for Yes; 0 for No.
    # Think very hard before turning this off!
    backup = 1

    # Where shall we keep it ?
    # Remember to back up this directory regularly!
    backup_dir = "/etc/lvm/backup"

    # Should we maintain an archive of old metadata configurations.
    # Use 1 for Yes; 0 for No.
    # On by default.  Think very hard before turning this off.
    archive = 1

    # Where should archived files go ?
    # Remember to back up this directory regularly!
```

```
    archive_dir = "/etc/lvm/archive"

    # What is the minimum number of archive files you wish to keep ?
    retain_min = 10

    # What is the minimum time you wish to keep an archive file for ?
    retain_days = 30
}

# Settings for the running LVM2 in shell (readline) mode.
shell {

    # Number of lines of history to store in ~/.lvm_history
    history_size = 100
}


# Miscellaneous global LVM2 settings
global {
    library_dir = "/usr/lib64"

    # The file creation mask for any files and directories created.
    # Interpreted as octal if the first digit is zero.
    umask = 077

    # Allow other users to read the files
    #umask = 022

    # Enabling test mode means that no changes to the on disk metadata
    # will be made.  Equivalent to having the -t option on every
    # command.  Defaults to off.
    test = 0

    # Default value for --units argument
    units = "h"

    # Whether or not to communicate with the kernel device-mapper.
    # Set to 0 if you want to use the tools to manipulate LVM metadata
    # without activating any logical volumes.
    # If the device-mapper kernel driver is not present in your kernel
    # setting this to 0 should suppress the error messages.
    activation = 1

    # If we can't communicate with device-mapper, should we try running
    # the LVM1 tools?
    # This option only applies to 2.4 kernels and is provided to help you
    # switch between device-mapper kernels and LVM1 kernels.
    # The LVM1 tools need to be installed with .lvm1 suffices
    # e.g., vgscan.lvm1 and they will stop working after you start using
    # the new lvm2 on-disk metadata format.
    # The default value is set when the tools are built.
    # fallback_to_lvm1 = 0

    # The default metadata format that commands should use - "lvm1" or
```

```
"lvm2".
    # The command line override is -M1 or -M2.
    # Defaults to "lvm1" if compiled in, else "lvm2".
    # format = "lvm1"

    # Location of proc filesystem
    proc = "/proc"

    # Type of locking to use. Defaults to local file-based locking (1).
    # Turn locking off by setting to 0 (dangerous: risks metadata
corruption
    # if LVM2 commands get run concurrently).
    # Type 2 uses the external shared library locking_library.
    # Type 3 uses built-in clustered locking.
    locking_type = 3

    # If using external locking (type 2) and initialization fails,
    # with this set to 1 an attempt will be made to use the built-in
    # clustered locking.
    # If you are using a customized locking_library you should set this to
0.
    fallback_to_clustered_locking = 1

    # If an attempt to initialize type 2 or type 3 locking failed, perhaps
    # because cluster components such as clvmd are not running, with this
set
    # to 1 an attempt will be made to use local file-based locking (type
1).
    # If this succeeds, only commands against local volume groups will
proceed.
    # Volume Groups marked as clustered will be ignored.
    fallback_to_local_locking = 1

    # Local non-LV directory that holds file-based locks while commands are
    # in progress.  A directory like /tmp that may get wiped on reboot is
OK.
    locking_dir = "/var/lock/lvm"

    # Other entries can go here to allow you to load shared libraries
    # e.g., if support for LVM1 metadata was compiled as a shared library
use
    #   format_libraries = "liblvm2format1.so"
    # Full pathnames can be given.

    # Search this directory first for shared libraries.
    #   library_dir = "/lib"

    # The external locking library to load if locking_type is set to 2.
    #   locking_library = "liblvm2clusterlock.so"
}

activation {
    # Device used in place of missing stripes if activating incomplete
volume.
```

```
    # For now, you need to set this up yourself first (e.g., with
'dmsetup')
    # For example, you could make it return I/O errors using the 'error'
    # target or make it return zeros.
    missing_stripe_filler = "/dev/ioerror"

    # How much stack (in KB) to reserve for use while devices suspended
    reserved_stack = 256

    # How much memory (in KB) to reserve for use while devices suspended
    reserved_memory = 8192

    # Nice value used while devices suspended
    process_priority = -18

    # If volume_list is defined, each LV is only activated if there is a
    # match against the list.
    #   "vgname" and "vgname/lvname" are matched exactly.
    #   "@tag" matches any tag set in the LV or VG.
    #   "@*" matches if any tag defined on the host is also set in the LV
or VG
    #
    # volume_list = [ "vg1", "vg2/lvol1", "@tag1", "@*" ]

    # Size (in KB) of each copy operation when mirroring
    mirror_region_size = 512

    # 'mirror_image_fault_policy' and 'mirror_log_fault_policy' define
    # how a device failure affecting a mirror is handled.
    # A mirror is composed of mirror images (copies) and a log.
    # A disk log ensures that a mirror does not need to be re-synced
    # (all copies made the same) every time a machine reboots or crashes.
    #
    # In the event of a failure, the specified policy will be used to
    # determine what happens:
    #
    # "remove" - Simply remove the faulty device and run without it.  If
    #            the log device fails, the mirror would convert to using
    #            an in-memory log.  This means the mirror will not
    #            remember its sync status across crashes/reboots and
    #            the entire mirror will be re-synced.  If a
    #            mirror image fails, the mirror will convert to a
    #            non-mirrored device if there is only one remaining good
    #            copy.
    #
    # "allocate" - Remove the faulty device and try to allocate space on
    #            a new device to be a replacement for the failed device.
    #            Using this policy for the log is fast and maintains the
    #            ability to remember sync state through crashes/reboots.
    #            Using this policy for a mirror device is slow, as it
    #            requires the mirror to resynchronize the devices, but it
    #            will preserve the mirror characteristic of the device.
    #            This policy acts like "remove" if no suitable device and
    #            space can be allocated for the replacement.
```

```
    #             Currently this is not implemented properly and behaves
    #             similarly to:
    #
    # "allocate_anywhere" - Operates like "allocate", but it does not
    #             require that the new space being allocated be on a
    #             device is not part of the mirror.  For a log device
    #             failure, this could mean that the log is allocated on
    #             the same device as a mirror device.  For a mirror
    #             device, this could mean that the mirror device is
    #             allocated on the same device as another mirror device.
    #             This policy would not be wise for mirror devices
    #             because it would break the redundant nature of the
    #             mirror.  This policy acts like "remove" if no suitable
    #             device and space can be allocated for the replacement.

    mirror_log_fault_policy = "allocate"
    mirror_device_fault_policy = "remove"
}


####################
# Advanced section #
####################

# Metadata settings
#
# metadata {
    # Default number of copies of metadata to hold on each PV.  0, 1 or 2.
    # You might want to override it from the command line with 0
    # when running pvcreate on new PVs which are to be added to large VGs.

    # pvmetadatacopies = 1

    # Approximate default size of on-disk metadata areas in sectors.
    # You should increase this if you have large volume groups or
    # you want to retain a large on-disk history of your metadata changes.

    # pvmetadatasize = 255

    # List of directories holding live copies of text format metadata.
    # These directories must not be on logical volumes!
    # It's possible to use LVM2 with a couple of directories here,
    # preferably on different (non-LV) file systems, and with no other
    # on-disk metadata (pvmetadatacopies = 0). Or this can be in
    # addition to on-disk metadata areas.
    # The feature was originally added to simplify testing and is not
    # supported under low memory situations - the machine could lock up.
    #
    # Never edit any files in these directories by hand unless you
    # you are absolutely sure you know what you are doing! Use
    # the supplied tool set to make changes (e.g., vgcfgrestore).

    # dirs = [ "/etc/lvm/metadata", "/mnt/disk2/lvm/metadata2" ]
#}
```

```
# Event daemon
#
# dmeventd {
    # mirror_library is the library used when monitoring a mirror device.
    #
    # "libdevmapper-event-lvm2mirror.so" attempts to recover from failures.
    # It removes failed devices from a volume group and reconfigures a
    # mirror as necessary.
    #
    # mirror_library = "libdevmapper-event-lvm2mirror.so"
#}
```